Net-X-Code API v7

User Guide

July 18, 2025

1 Introduction....8
1.1 Conventions....8
1.2 About Net-X-Code....9
1.3 System Requirements....10
1.3.1 Recommended Environment....10
2 Theory of Operation....11
2.1 Typical Applications....13
2.1.1 Net-X-Code IP Video Capture....13
2.1.2 Net-X-Code IP Video Streaming....13
2.1.3 Partial File Restore, Conversion and Proxy....14
2.2 Changes In Version 7 API:....15
2.3 Command And Return Structure/Format....15
2.4 Command Elements....16
2.5 Return Elements....17
2.6 Command – general....18
2.6.1 Active....18
2.6.2 TestFunc....18
2.6.3 netxbase=version....18
2.6.4 Return – bad/malformed request....19
2.6.5 Return – shutdown....19
2.6.6 Get A Config Setting....19
2.6.7 Set A Config Setting....20
2.7 Command - get....20
2.7.1 Client Count....20
2.7.2 Client IP....21
2.7.3 Number Of Groups....21
2.7.4 Attributes/Request Info....22
3 Stream States....23
3.1.1 previewenabled....31
3.1.2 preview....31
3.1.3 pts....32
3.1.4 datarate....33
3.1.5 property....34
3.1.6 status....37
3.1.7 streamstate....38
3.1.8 state....40
3.1.9 xmldisable....41
3.1.10 autostart....42
3.1.11 getDiscontinuities....42
3.1.12 getCopyInOut....44
3.1.12.1 Tape PFR using the command line....46
3.1.13 getFirstAndLastTimecode....47
3.2 Command – Copy, Convert and PFR....48
3.2.1 set – initiate copy/convert/pfr....48
3.2.2 set/get – cardinfo (camera card clip lists)....51
3.2.3 set/get – clipfiles (files included in a clip)....54
3.2.4 set/get – metdata (file information and metadata in XMP format)....63
3.2.4.1 Parameters:....72
3.2.5 copylimit....76
3.2.6 get - status/completion....77

3.3 Typical Proxy, Convert and PFR Session....84
3.3.1 Scenario 1 – full access....84
3.3.2 Scenario 2 – tape restore....85
3.3.3 Scenario 3 – cloud restore....85
3.3.4 Scenario 4 – in line conversions....85
3.4 PFR File Best Practices....86
3.4.1 Self Contained....86
3.4.2 Multiple Stream Files....86
3.4.3 Multiple Files Per Stream....86
3.4.4 Avid OP-Atom – Special Case....87
3.4.5 Sequences....87
3.5 Command - set....87
3.5.1 start....88
3.5.2 startstream....88
3.5.3 stop....88
3.5.4 restart....89
3.5.5 xmldisable....90
3.5.6 autostart....90
3.5.7 add....92
3.5.8 remove....93
3.5.9 starttc/endtc....93
3.5.10 delete....94
3.5.11 Proxymode....94
3.5.12 proxyfiletype....95
3.5.13 proxycompression....95
3.5.14 proxydatarate....96
3.5.15 tsenable....96
3.5.16 tsdir....96
3.5.17 tsfile....97
3.6 SETTINGS....97
3.7 Media File Scanner Command Set....103
3.7.1 Getting a List of Scanning Instances....103
3.7.2 Adding new Media File Scanner (newscandir=)....104
3.7.3 Deleting An Instance....105
3.7.4 Clearing Errors For An Instance....106
3.7.5 Set/Get Directory Path or Database....106
3.7.6 Skipping Extensions or Paths....108
3.7.6.1 Skip Extensions....108
3.7.6.2 Skip Strings (Paths)....109
3.7.6.3 Skip Strings (Name)....109
3.7.7 Starting a Scan....110
3.7.8 Monitoring a Scan....111
3.7.8.1 gettotaldirectories....111
3.7.8.2 gettotalfiles....112
3.7.8.3 gettotalclips....112
3.7.8.4 getprocesseswaiting....112
3.7.8.5 getoperation....112
3.7.9 Media File Scanner Configure....112
3.7.9.1 Max Translations....112

3.7.9.2 Max MD5 Translations....113
3.7.9.3 Make MD5....113
3.7.9.4 Make Picon....114
3.7.9.5 Make Proxy....119
3.7.9.6 Set Mezzanine....127
3.7.9.7 Make XMP....134
3.8 Creating an MD5....136
3.9 Scheduling....137
3.9.1 Month View....137
3.9.2 Week View....138
3.9.3 Day View....139
3.9.4 Create Event....140
3.9.5 Generate a new....143
3.9.6 Get a list of events....144
3.10 QC/QA Processing....145
3.10.1 To compare two files....145
3.10.2 To analyze the audio and video in a file....147
3.10.3 To validate a file against a pre-made profile....148
3.10.4 To compare a file against another file for basic parameters....148
3.11 Watch Folders....149
3.12 Command - callstate....149
3.13 Command – close/restart....152
3.14 Command – Thumbnail/JPEG/Picons....153
3.14.1 Live Confidence Monitoring....153
3.14.2 preview....153
3.14.3 mpreview....154
3.14.4 Create JPEGs On Disk....154
3.14.5 Picon....154
3.14.6 Return A JPEG From A File....155
3.14.7 Picon....155
3.15 NetXSDI – MediaCMD....157
3.15.1 SetCurChannel....157
3.15.2 Play....158
3.15.3 PlayAtSpeed....158
3.15.4 PlayFromTo....158
3.15.5 LoadClip....158
3.15.6 PlayClip....159
3.15.7 PlayClipFromTo....159
3.15.8 FastForward....159
3.15.9 FastRewind....159
3.15.10 Pause....160
3.15.11 Seek....160
3.15.12 SeekRelative....160
3.15.13 Stop....160
3.15.14 Record....161
3.15.15 RecordFromTo....161
3.15.16 RecordStop (prepare record)....161
3.15.17 SetRecordPresets....161
3.15.18 Eject....162
3.15.19 Transfer....162

3.15.20 Update Status....162
3.15.21 GetState....163
3.15.22 GetSpeed....163
3.15.23 GetPosition....164
3.15.24 GetLastMS....164
3.15.25 GetStart....165
3.15.26 GetEnd....165
3.15.27 GetClipName....165
3.15.28 GetCurTC....165
3.15.29 GetCurState....166
3.15.30 GetNextClip....166
3.15.31 GetClipInfo....166
3.15.32 EDLGetEdit....167
3.15.33 Insert....167
3.15.34 Blank....168
3.15.35 Delete....168
3.15.36 Trim....168
3.15.37 ValueSupported....169
3.15.38 GetValue....169
3.15.39 SetValue....169
3.15.40 GetValue/SetValue Metadata....169
3.15.40.1 Strings....170
3.15.40.2 Numeric....173
3.15.40.3 Read Only....175
3.15.41 Get/SetClipMode....176
3.15.42 Get/SetTCType....177
3.15.43 Get/SetTCSource....177
3.15.44 Get/SetAudioInput....177
3.15.45 Get/SetAudioInputLevel....178
3.15.46 Get/SetAudioOutput....178
3.15.47 Get/SetAudioOutputLevel....179
3.15.48 Get/AudioPeakRMS....179
3.15.49 Get/SetVideoInput....179
3.15.50 Get/SetVideoInputSetup....180
3.15.51 Get/SetVideoInputVideo....180
3.15.52 Get/SetVideoInputHue....181
3.15.53 Get/SetVideoInputChroma....181
3.15.54 Get/SetVideoTBCSetup....181
3.15.55 Get/SetVideoTBCVideo....181
3.15.56 Get/SetVideoTBCHue....182
3.15.57 Get/SetVideoTBCChroma....182
3.16 Full MediaCmd AJAX/XML Access....182
3.16.1 sdicmd main commands....186
3.16.2 Basic Command (sdicmd)....187
3.16.3 sdicmd Examples....189
3.16.4 Dealing with Picon Images....189
3.17 Special XML Access Commands....191
4 Net-X-Time-Code....193
4.1.1 Set A Variable....194
4.1.2 Get A Variable....194

4.1.3 Get/Set Variables....194
4.1.4 Start Capturing Time Code....195
4.1.5 Stop Capturing Time Code....195
4.2 Net-X-Copy Command Line....195
4.3 Discontinuity Sources In Net-X-Code....200
4.3.1 Discontinuity Reporting....201
4.4 Error Returns....202
4.5 Error Returns – Net-X-Copy....203
4.6 Error Returns – QC....204
4.7 Error Returns – Net-X-SDI....205
4.8 ACK/ACKC/ACKR File Format....207
4.9 Discontinuity Handling in Net-X-Copy....210
4.10 Configuration....210
4.10.1 Net-X-Base (NetXCode)....210
4.10.2 Net-X-Cmd....211
4.10.3 Net-X-Copy....211
4.10.4 Net-X-Time-Code....212
4.11 General Config: config.xml....212
4.11.1 Windows:....212
4.11.2 Linux:....212
4.11.3 macOS:....212
4.12 Automatic Proxy Generation Setup....212
4.13 Multirate TS/HLS and MP4/DASH setup....213
4.14 System Setup....214
4.14.1 Linux (CentOS/Red Hat) - Network....215
4.14.2 Windows - Network....215
4.14.3 macOS - Network....216
4.14.4 Linux – Required Packages....216
4.14.5 Linux – SysLog Output....216
5 Copyrights and Trademark Notices....218
5.1 General....218
5.2 GNU LESSER GENERAL PUBLIC LICENSE....227
5.2.1.1 0. Additional Definitions....227
5.2.1.2 1. Exception to Section 3 of the GNU GPL....227
5.2.1.3 2. Conveying Modified Versions....227
5.2.1.4 3. Object Code Incorporating Material from Library Header Files....228
5.2.1.5 4. Combined Works....228
5.2.1.6 5. Combined Libraries....229
5.2.1.7 6. Revised Versions of the GNU Lesser General Public License....229
5.3 MPEG Disclaimers....230
5.3.1 MPEGLA MPEG2 Patent....230
5.3.2 MPEGLA MPEG4 VISUAL....230
5.3.3 MPEGLA AVC....230
5.3.4 MPEG4 SYSTEMS....230
5.4 Drastic Technologies Limited Warranty and Disclaimers....231
5.4.1 Warranty Remedies....231
5.4.2 Software Updates....231
5.4.3 Restrictions and Conditions of Limited Warranty....231
5.4.4 Limitations of Warranties....232

5.4.5 Damages..... 232

1 Introduction

This manual is for the Net-X-Code 7.x software development kit from Drastic Technologies, Ltd.

1.1 Conventions

This manual assumes the following:

That the user knows how to operate a mouse and keyboard and perform the basic functions of Microsoft Windows, macOS or Linux operating system.

That the user is familiar with the creative software in use.

That the user has access to technicians capable of placing the device on the network and setting up any SAN systems if necessary.

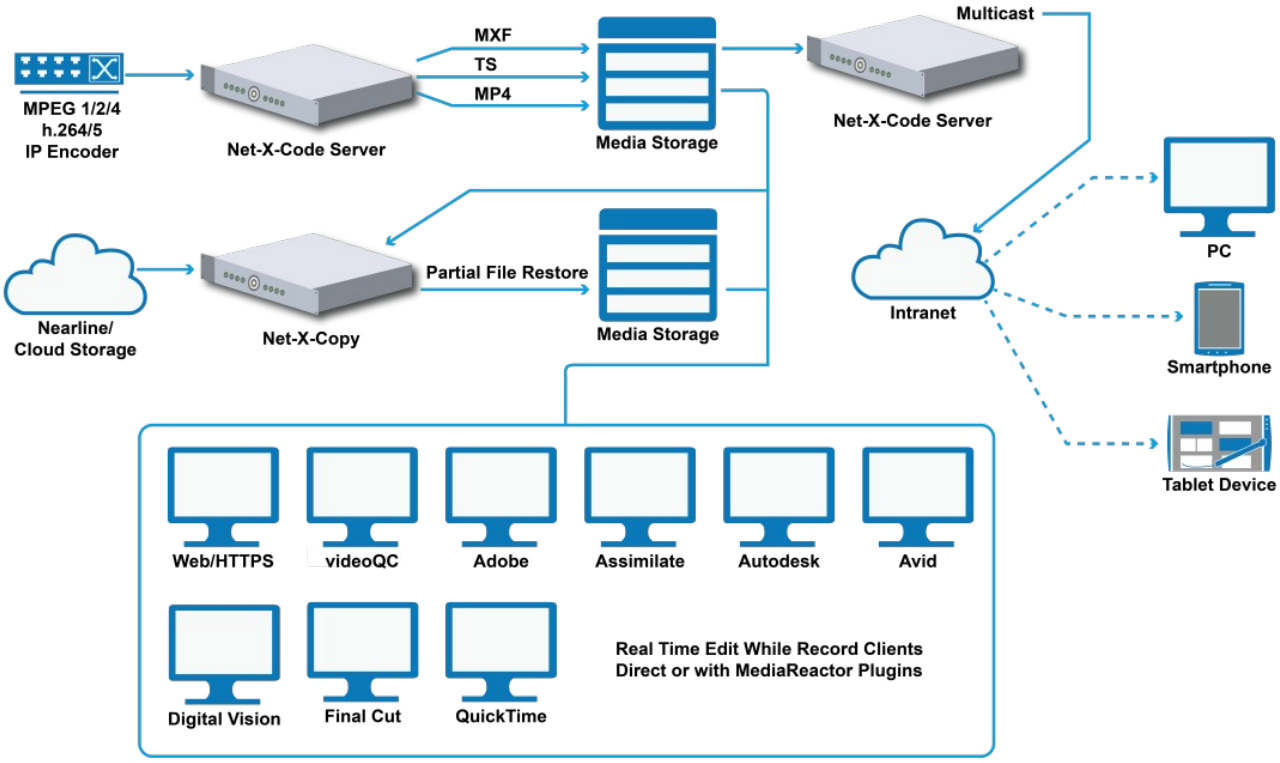
The name of a control or display present on the interface will be displayed in **bold** text.

Where a portion of the manual is referred to the name of section mentioned will be displayed in *italics*.

Certain images in this document may have been grayed out where it is useful or necessary to place indicator marks to show specific controls or displays above a darker background.

1.2 About Net-X-Code

Net-X-Code is a series of interconnected applications (Net-X-Code, Net-X-Cmd, Net-X-Copy, Net-X-Streamer, videoQC and MediaReactor Workstation plugins) for IP based video capture, streaming, and conversion. The various applications auto-detect and join user-defined enterprise groups on one or more servers within a network. Once connected, the controller/user can configure the system in real time via the HTTP/RESTful interface. The major components are diagrammed and listed below:



For the latest Net-X-Code information, please see:

http://www.net-x-code.com

1.3 System Requirements

1.3.1 Recommended Environment

The Net-X-Code software development kit should be installed on a modern system capable of performing common computer functions.

Minimum Hardware Platform
Post 2000 multi core processor capable of running the host application.

Recommended Hardware Platform
Recommend hardware from the host application vendor. Multi cores and OpenCL, Cuda and shader GPUs will be used if available, but are not absolutely required.

Demo downloads of Drastic software are available for the user to test their application and to confirm their workflow. Faster and more powerful hardware will provide better performance. In some cases, specific hardware will be required in order to enable resource-intensive, advanced or optional features.

2 Theory of Operation

Net-X-Code is a distributed capture and conversion system. It can be run on one or more servers and be controlled from one, central interface. This section of the manual will give an overview of how the various parts of Net-X-Code interact to make it easier to design deployments and implement controllers using the API described in the next section.

Net-X-Code is made up of a number of servers, programs, and plugins:

Net-X-Cmd: This component provides the central connection for all the other components. It uses a Bonjour-like protocol to auto-sense components within its group in the network, and provides the HTTP/RESTful/HTML API

Net-X-Base: is the central command for all the Net-X-Cmd servers in a Net-X-Code cluster. There must be one Net-X-Base to make all the components of a cluster work together, though some workflows use one per server.

Net-X-Code: Provides capture from network IP video sources to TS, MP4, fMP4, ISM, MXF, etc. files. A Net-X-Code server can capture 1 or more groups of up to 4 streams per group. Files can also include DASH, HLS, and Smooth Streaming sidecar files. These recorded files can be stored locally or anywhere else on the network.

Net-X-Server: Can take recording or pre-recorded network IP video streams from disk anywhere in the network, and re-stream them via SMPTE 2110, SMPTE 2022=6, RTP, or UDP back out to the network

Net-X-Copy: Is both a real time video translator, real time clipping engine and partial file restore system. Any recording stream can be used as a source while it is still recording, or near line/tape backup files can be restored, only accessing the part of the file required for the restore.

Net-X-Player: provides an HTML5 based web player that can play time coded, frame accurate MP4 files, including playback while they are still recording. Net-X-Player can also generate clipping commands to use on live, or recorded streams.

Media File Scanner: is an optional plugin that can scan local, network, Amazon S3, Google Cloud or other file repositories to identify and organize your media. The scan works by creating media 'clips' that are the actual audio/video/metadata by combining and stitching together associated audio and video files. The result is a set of actual media files, rather than just every file, that can be used to extract metadata, make proxies, generate thumbnails and push all the information into a database or your choice of MAM.

videoQC: Video preview is available from on disk, live recording and network video sources. videoQC provides video waveform, vectorscope, histogram, chromaticity, and metadata displays, along with clipping and conversions.

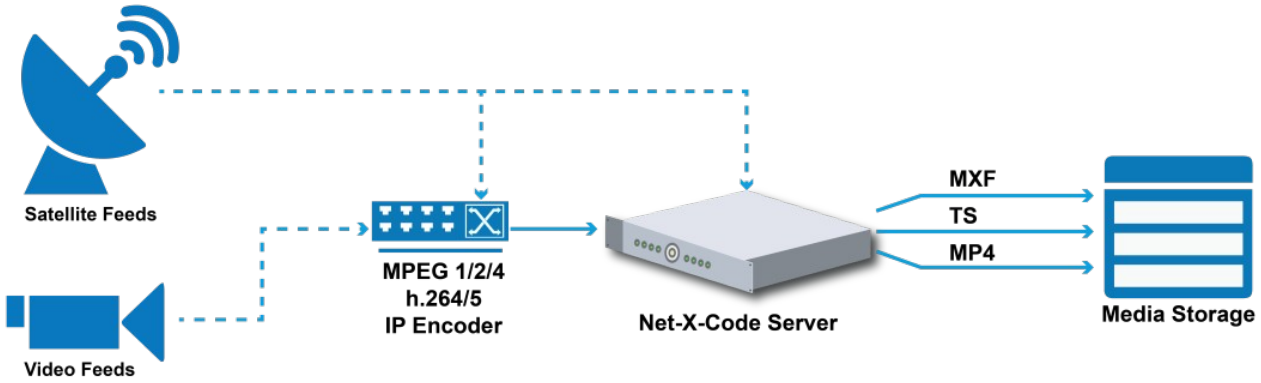
MediaReactor Workstation: These plugins allow professional editing and finishing systems to access all the file types created by the Net-X-Code system, and access the real time recording files while they are still growing. MediaReactor Workstation is built into software

like Assimilate and Nucoda, and available as an option for other systems like Adobe, Avid, Autodesk, Quantel, Final Cut, and even QuickTime compatible apps.

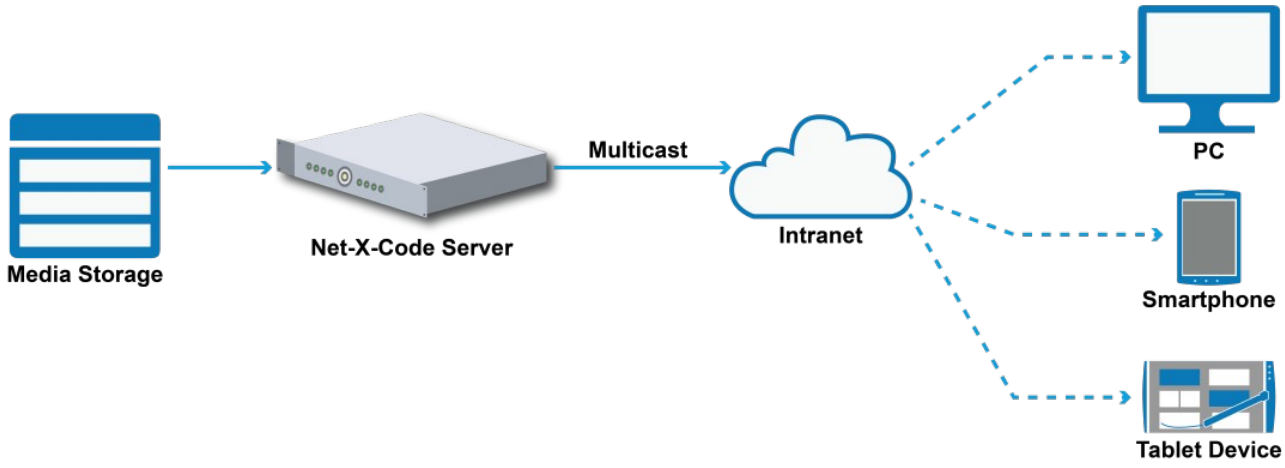
ccConvert provides embedding, extraction, and transcoding features for a wide range of closed caption formats. Command line functionality is available for automatable conversions. Included ccReview plays captions within a video file, sidebar caption files with the same name as a video file, and supports playing arbitrary caption files over video, as well as over black.

2.1 Typical Applications

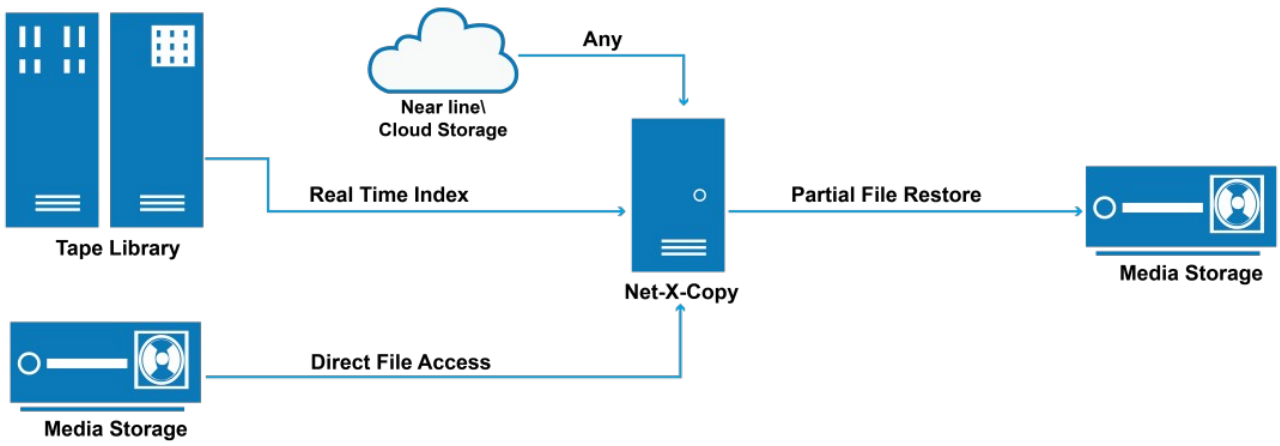
2.1.1 Net-X-Code IP Video Capture



2.1.2 Net-X-Code IP Video Streaming



2.1.3 Partial File Restore, Conversion and Proxy



2.2 Changes In Version 7 API:

- All XML returns use “UTF-8”. Previous APIs used “ISO 8859-1”
- All returns include a <result> key which has a 0 for success or any other number for failure
- All returns include a <resultstring> with a short result message
- All multi element returns (like lists of <clip> returns) include an “index=#” attribute
- ‘message’ and other string return attributes are now in <resultstring>
- All progress returns are now based on 0..10000 (QC returns were 0..100)
- Successful operations’ progress will end at 10000. Errored operations will progress to -1
- There is a “completed” element now which indicates the operation has moved to the completed list, when asking for task status on the in process plug completed list
- Picon generation operation can now be searched for and use the completed list
- JSON is now officially supported (with ‘/netxjson?’) for most commands (see exceptions in this manual)
- Fixed typo of ‘Muti’ to ‘Multi’ for DASH, HLS, and MultiOutput

2.3 Command And Return Structure/Format

Commands are sent as requests to Net-X-Base’s integrated server. By default, this server exists at port 1080 on the server’s external IP. The beginning of the command will always be netx?request=get& or netx?request=set& followed by the request. Please note that as soon as the command is validated, a positive response is returned. This keeps the server running quickly for status updates and multiple commands. For commands that may take some time, the result of the command should be checked and waited for by monitoring its status. For instance, once a create group command has been sent, requesting the status for that group's key name will let the caller know when it is ready for more commands. To get the initial status of the system, the recommended order of events is:

- Request client count
- Request client address(s) from 0 - client count
- Request group count from client
- Request info (channel count is fixed at 3)

Example: Determining the number of clients.

A basic request for client count would look like this:

```
http://127.0.0.1:1080/netx?request=get&client=count
```

The response will be an XML buffer like this, or a JSON buffer if using ‘netxjson’:

<?xml version="1.0" encoding="UTF-8"?>
<client count="1">
<result>0</result>
<resultstring>returning client count</resultstring>
</client>
	{
	"count": "1",
	"result": "0",
	"resultstring": "returning client count"
	}

NOTE: for the rest of the returns in this manual, we will not include the ‘<?xml version="1.0" encoding="UTF-8"?>’ start, but it will always be returned if returning XML.

2.4 Command Elements

The commands are sent via HTTP/REST as an HTTP request string. The commands must start with one of two strings:

- netx? - Command will return an XML response
- netxjson? - Command will return a JSON response

Depending on whether it is a command or a request for information, this is followed by “request=get” or “request=set” without quotes:

Basic command start
http://127.0.0.1:1080/netx?request=get
http://127.0.0.1:1080/netx?request=set

Command Parameters

The parameters may be applied to the command in any order. A command may not use any parameter more than once in a single command, nor may channel sets be combined with a general group command (more on this in the group command section). The available command parameters are:

- client=<ip of client, count, address> Used to specify the client server machine the request or command is intended for. For requests, if 'count' is specified it will return the number of clients. Also for requests, if 'address' is specified, then it will return the client address.
- group=<#, KEY, all, count> Used to specify the group of the machine the request or command will act on. The group parameters can have four forms:
  - # the group number, as returned in the list

- KEY the group name as set when created, or later, by the caller
- all specify that all channels should be returned or acted upon
- count to request the total number of groups
    - index=# specifies the index number for the command or request.
    - channel=<#, all> will specify the channel within the group the command or request requires. If it is set to 'all' then it will return a request for all channels in that group.
    - value=<#> set a value (numeric) for a command
    - command=<cmd> is used when sending a command. More information on this is available below. The possible commands are:
    - start – start a capture
    - stop – stop a capture
    - restart – restart a capture
    - add – add a new group
    - remove – remove an existing group
    - delete – delete files on the disk. List files with &file=<file path name> pairs.

2.5 Return Elements

All XML returns from the Net-X-Code REST API will always start with <?xml version="1.0" encoding="UTF-8"?> followed by at least one <key>. That first <key> will always include two keys within it:

<result>0/result> - which will contain a 0 for success, and any other number for failure. The most common will be -1 for an unknown failure. If there are extended error numbers available (like errno or other values) then a failure may produce a negative or positive number.
<resultstring>command=success</resultstring> - a short message describing the results. It will usually be fairly general, but may include specifics, especially if the result is not 0.
Common return keys include:

- active, testfunc, request,
- netxbase, copy, client, clients, errors, count,
- group, getdiscontinuities, getcopyinout, copyinoutinfo,
- getfirstandlasttimecode

<something>
<result>0/result>
<resultstring>returning client count</resultstring>
</something>

2.6 Command – general

2.6.1 Active

http://127.0.0.1:1080/netxjson?active=1	

<?xml version="1.0" encoding="UTF-8"?>
<active>
<result>0</result>
<resultstring>active=true</resultstring>
</active>
	{
	"result": "0",
	"resultstring": "active=true"
	}

Is the REST API available. May not return if the service is not running, otherwise it should return a result of 1 and a resultstring of active=true.

2.6.2 TestFunc

http://127.0.0.1:1080/netx?testfunc=createpath&pathtocreate	

<testfunc>
<result>22</result>
<resultstring>unknown test function</resultstring>
</testfunc>
	{
	"result": "22",
	"resultstring": "unknown test function"
	}

TestFunc contains test or utility functions. Createpath can be used to make a multi level path on the API server. This, and other TestFunc are reserved for internal use. Do not use, as it can cause the API to crash.

2.6.3 netxbase=version

http://127.0.0.1:1080/netxjson?netxbase=version	
<?xml version="1.0" encoding="UTF-8"?> <netxbase version="v7.0.0.1" address="192.168.50.100"> <result>0</result> <resultstring>nexbase version	{ "address": "192.168.50.100", "result": "0", "resultstring": "nexbase version returned", "version": "v7.1.0.3"

returned</resultstring> </netxbase>	}

This request returns the version and the IP address of the Net-X-Base REST API.

2.6.4 Return – bad/malformed request

If the system is shutting down, then all commands will return the following

```
<request netxbaseip="192.168.50.100" type="">
  <result>0</result>
  <resultstring>failure</resultstring>
</request>
```

2.6.5 Return – shutdown

If the system is shutting down, then all commands will return the following

```
<request>
  <result>0</result>
  <resultstring>ignored – shut down in progress</resultstring>
</request>
```

2.6.6 Get A Config Setting

Return the value of a setting from the config.xml from a server.

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&configaction=getvalue&configkey=VVW\Internal0&configsetting=AESEBU&configtype=dword

<config>
<configkey>VVW\Internal0</configkey>
<configitem>AESEBU</configitem>
<configvalue>0</configvalue>
<configtype>dword</configtype>
</config>
	{
	"configitem" : "AESEBU",
	"configkey" : "VVW\\Internal0",
	"configtype" : "dword",
	"configvalue" : "0"
	}

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&configaction=getvalue&configkey=VVW\Internal0&configsetting=Dir&configtype=string

<config>
  <configkey>VVW\Internal0</configkey>
  <configitem>Dir</configitem>
  <configvalue>\\192.168.101.100\
AvidMedia\Avid MediaFiles\</configvalue>
  <configtype>string</configtype>
</config>
	{
	  "configitem":"Dir",
	  "configkey":"VVW\\Internal0",
	  "configtype":"string",
	  "configvalue":"\\\\192.168.101.100\\
	AvidMedia\\Avid MediaFiles\\"
	}

2.6.7 Set A Config Setting

Set the value of a setting from the config.xml from a server. Note: the request is still “request=set”, but the configaction is now “configaction=setvalue”.

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&configaction=setvalue&configkey=VVW\Internal0&configsetting=AESEBU&configtype=dword&configvalue=0

<config>
  <configkey>VVW\Internal0</configkey>
  <configitem>AESEBU</configitem>
  <configvalue>0</configvalue>
  <configtype>dword</configtype>
</config>
	{
	  "configitem" : "AESEBU",
	  "configkey" : "VVW\\Internal0",
	  "configtype" : "dword",
	  "configvalue" : "0"
	}

2.7 Command - get

To get the status of all the servers, groups and channels, the following order is recommended:

1. Request client count
2. Request client address(es) from 0 client count
3. Request group count from client
4. Request info (channel count is fixed at 3)

2.7.1 Client Count

Request the client count client=count

http://127.0.0.1:1080/netx?request=get&client=count

<client count="1">
<result>0</result>
<resultstring>returning client count</resultstring>
</client>
	{
	"count": "1",
	"result": "0",
	"resultstring": "returning client count"
	}

2.7.2 Client IP

Request the first client IP address client=ip&index=#. *Note that the address and the nextbaseip might be the same when the base and cmd run on the same computer

http://127.0.0.1:1080/netxjson?request=get&client=address&index=0

<client netxbaseip="192.168.50.100" address="192.168.100.229" version="v7.1.0.1">
<result>0</result>
<resultstring>returning client</resultstring>
</client>
	{
	"address": "192.168.100.229",
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "returning client",
	"version": "v7.1.0.3"
	}

2.7.3 Number Of Groups

Request the number of groups client=ip&group=count. The client can be its numerical value (0..n), or its IP address.

http://127.0.0.1:1080/netx?request=get&client=0&group=count

<client netxbaseip="192.168.50.100" address="192.168.100.229" version="v7.1.0.1" groups="3" autostart="0">
<group count="3"/>
<result>0</result>
<resultstring>group returned</resultstring>
</client>
	{
	"address": "192.168.100.229",
	"autostart": "0",
	"group": "",
	"groups": "3",
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "group returned",
	"version": "v7.1.0.3"
	}

2.7.4 Attributes/Request Info

Request attributes client=ip&group=#&channel=#&value=<setting> where ‘setting’ can be:

- **name** - the target file name
- **capturefile** – the full name and path of the actual capture file
- **address** - the address for the capture
- **port** - the port for the capture
- **directory** - the target directory
- **rectc** - type or record time code capture
- **type** - the target file type (.mp4, .fmp4, .mov, .ism, .mxf, .ts)
- **state** - the current state
- **ms** - the last millisecond count
- **bytes** - the last byte count
- **uuid** - the channel’s uuid
- **metadata** – the metadata string from the stream
- **frames** – the number of video frames captured
- **tc** - the last gop start captured time code
- **timecode** – the last captured time code
- **ub** – the current captured userbits
- **firsttc** – the first time code captured for a stream
- **firstub** – the first userbits captured for a stream
- **tctype** - channel’s time code type
    - 1 = 24.0 fps
    - 2 = 30 fps (non drop frame 30)
    - 4 = 29.97 fps (drop frame 30)
    - 8 = 25.0 fps
    - 16 = 50.0 fps
    - 32 = 59.94 fps (drop frame 60)
    - 64 = 60.0 fps (non drop frame 60)
- **starttc** - capture start time
- **endtc** - capture end time
- **all** (or "**-1**") - return all of the above
- **previewenabled** – retrieve the current state of preview frames
- **preview** – retrieve a JPEG preview embedded in the XML response of the last known valid I-Frame
- **mpreview** – retrieve a JPEG preview of the last known valid I-Frame as raw image info
- **property** – retrieve the stream video/audio properties
- **status** – retrieve a list of status messages

3 Stream States

The possible stream states are:

NetXState_Idle = 0
"Idle"
NetXState_Active = 1
"Active"
NetXState_SourceConnectionPending = 2
"Connection Pending"
NetXState_SourceClosing = 3
"Source Closing"
NetXState_SourceDisconnected = 4
"Source Disconnected"
NetXState_SourceCapturePending = 5
"Source Capture Pending"
NetXState_SourcePending = 6
"Waiting For Source"
NetXState_Error = 7
"Error"
NetXState_NotFound = 8
"Not Running"
NetXState_Unknown = -2
"Unknown"

Info about all groups and/or all channels can be returned at once. Setting "all" or a specific value for any of the parameters group="", channel="", value="" will return different results. Here are some examples:

Requesting a single parameter from one group and one channel.

http://127.0.0.1:1080/netxjson?request=get&client=0&group=0&channel=0&value=name

<client netxbaseip="192.168.50.100" address="192.168.100.229" version="v7.1.0.3" groups="3" autostart="0">
<group index="0" key="newgroupname" version="v7.1.0.3" autostart="1">
<channel index="0">
<name>Channel1</name>
</channel>
</group>

{
"address": "192.168.100.229",
"autostart": "0",
"group0": {
"autostart": "1",
"channel0": {
"index": "0",
"name": "Channel1"
},

```
  <result>0</result>
  <resultstring>group returned</resultstring>
</client>
```

```
    "index": "0",
    "key": "newgroupname",
    "version": "v5.1.0.13"
  },
  "groups": "3",
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "group returned",
  "version": "v7.1.0.3"
}
```

http://127.0.0.1:1080/netx?request=get&client=0&group=0&channel=0&value=timecode

```
<client netxbaseip="192.168.50.100"
address="192.168.100.229" version="v7.1.0.3"
groups="3" autostart="0">
  <group index="0" key="DemoCapture"
version="v5.1.0.13" autostart="1">
    <channel index="0">
      <timecode
string="00:00:00;00">0</timecode>
    </channel>
  </group>
  <result>0</result>
  <resultstring>group returned</resultstring>
</client>
```

```
{
  "address": "192.168.100.229",
  "autostart": "0",
  "group0": {
    "autostart": "1",
    "channel0": {
      "index": "0",
      "timecode": {
        "string": "00:00:00;00",
        "value": "0"
      }
    },
    "index": "0",
    "key": "DemoCapture",
    "version": "v5.1.0.13"
  },
  "groups": "3",
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "group returned",
  "version": "v7.1.0.3"
}
```

http://127.0.0.1:1080/netx?request=get&client=0&group=0&channel=0&value=all

```
<client netxbaseip="192.168.50.100"
address="192.168.100.229" version="v7.1.0.3"
groups="3" autostart="0">
  <group index="0" key="DemoCapture"
version="v5.1.0.13" autostart="1">
```

```
{
  "address": "192.168.100.229",
  "autostart": "0",
  "group0": {
    "autostart": "1",
```

```
   <channel index="0">
     <name>HBR-Live</name>
     <capturefile/>
     <enabled>1</enabled>
     <address>239.254.30.30</address>
     <uuid/>
     <port>5004</port>
     <protocol>RTP</protocol>
     <rectc>1</rectc>
     <directory>D:\drastic.trunk\applications\web\
netx\video\live</directory>
     <type>.mxf-op1a</type>
     <capturecc>-1</capturecc>

<streamaddress>239.254.30.30</streamaddress
>
     <streamport>5004</streamport>
     <streamfile>na</streamfile>
     <streamenabled>1</streamenabled>
     <streamduration>0</streamduration>
     <previewenabled>0</previewenabled>
     <mp4proxymode>0</mp4proxymode>
     <mp4proxyfcc>1635148593</mp4proxyfcc>

<mp4proxydatarate>2000000</mp4proxydatarate
>

<mp4proxyscaledown>2</mp4proxyscaledown>
     <ms>0</ms>
     <bytes>0</bytes>
     <state desc="Not Running">8</state>
     <metadata/>
     <tc string="00:00:00;00">0</tc>
     <metatcpid>4294967294</metatcpid>
     <bcdtc/>
     <timecode
string="00:00:00;00">0</timecode>
     <firsttc string="00:00:00;00">0</firsttc>
     <frames string="00:00:00;00">0</frames>
     <firstub>0</firstub>
     <tctype string="29.97">4</tctype>
     <bcdub/>
```

```
   "channel0": {
     "address": "239.254.30.30",
     "bcdtc": "",
     "bcdub": "",
     "bytes": "0",
     "capturecc": "-1",
     "capturefile": "",
     "directory": "D:\\drastic.trunk\\applications\\
web\\netx\\video\\live",
     "enabled": "1",
     "endtc": "-1",
     "firsttc": {
       "string": "00:00:00;00",
       "value": "0"
     },
     "firstub": "0",
     "frames": {
       "string": "00:00:00;00",
       "value": "0"
     },
     "index": "0",
     "metadata": "",
     "metatcpid": "4294967294",
     "mp4proxydatarate": "2000000",
     "mp4proxyfcc": "1635148593",
     "mp4proxymode": "0",
     "mp4proxyscaledown": "2",
     "ms": "0",
     "name": "HBR-Live",
     "port": "5004",
     "previewenabled": "0",
     "protocol": "RTP",
     "rectc": "1",
     "starttc": "-1",
     "state": {
       "desc": "Not Running",
       "value": "8"
     },
     "streamaddress": "239.254.30.30",
     "streamduration": "0",
     "streamenabled": "1",
     "streamfile": "na",
```

<table>
<tr>
<td>

```
    <userbits>0</userbits>
    <starttc>-1</starttc>
    <endtc>-1</endtc>
    <vitctimecode>3452816845</vitctimecode>
    <vitcub>3452816845</vitcub>
   </channel>
  </group>
  <result>0</result>
  <resultstring>group returned</resultstring>
</client>
```

</td>
<td>

```
    "streamport": "5004",
    "tc": {
     "string": "00:00:00;00",
     "value": "0"
    },
    "tctype": {
     "string": "29.97",
     "value": "4"
    },
    "timecode": {
     "string": "00:00:00;00",
     "value": "0"
    },
    "type": ".mxf-op1a",
    "userbits": "0",
    "uuid": "",
    "vitctimecode": "3452816845",
    "vitcub": "3452816845"
   },
   "index": "0",
   "key": "DemoCapture",
   "version": "v5.1.0.13"
  },
  "groups": "3",
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "group returned",
  "version": "v7.1.0.3"
}
```

</td>
</tr>
</table>

Or all settings for all groups and all channels

http://127.0.0.1:1080/netx?request=get&client=0&group=all&channel=all&value=all

<table>
<tr>
<td>

```
<client netxbaseip="192.168.50.100"
address="192.168.100.229" version="v7.1.0.1"
groups="3" autostart="0">
  <group index="0" key="DemoCapture"
version="0" autostart="1">
   <channel index="0">
    <name>HBR-Live</name>
    <capturefile/>
    <enabled>1</enabled>
```

</td>
<td>

```
{
  "address": "192.168.100.229",
  "autostart": "0",
  "group0": {
   "autostart": "1",
   "channel0": {
    "address": "239.254.30.30",
    "bcdtc": "",
    "bcdub": "",
```

</td>
</tr>
</table>

```
    <address>239.254.30.30</address>
    <uuid/>
    <port>5004</port>
    <protocol>RTP</protocol>
    <rectc>1</rectc>
    <directory>D:\drastic.trunk\applications\web\
netx\video\live</directory>
    <type>.mxf-op1a</type>
    <capturecc>-1</capturecc>
    <streamaddress>na</streamaddress>
    <streamport>0</streamport>
    <streamfile>na</streamfile>
    <streamenabled>0</streamenabled>
    <streamduration>0</streamduration>
    <previewenabled>1</previewenabled>
    <mp4proxymode>0</mp4proxymode>
    <mp4proxyfcc>1635148593</mp4proxyfcc>

<mp4proxydatarate>2000000</mp4proxydatarate
>

<mp4proxyscaledown>2</mp4proxyscaledown>
    <ms>0</ms>
    <bytes>0</bytes>
    <state desc="Not Running">8</state>
    <metadata/>
    <tc string="00:00:00;00">0</tc>
    <metatcpid>4294967294</metatcpid>
    <bcdtc/>
    <timecode
string="00:00:00;00">0</timecode>
    <firsttc string="00:00:00;00">0</firsttc>
    <frames string="00:00:00;00">0</frames>
    <firstub>0</firstub>
    <tctype string="29.97">4</tctype>
    <bcdub/>
    <userbits>0</userbits>
    <starttc>-1</starttc>
    <endtc>-1</endtc>
    <vitctimecode>3452816845</vitctimecode>
    <vitcub>3452816845</vitcub>
  </channel>
```

```
    "bytes": "0",
    "capturecc": "-1",
    "capturefile": "",
    "directory": "D:\\drastic.trunk\\applications\\
web\\netx\\video\\live",
    "enabled": "1",
    "endtc": "-1",
    "firsttc": {
      "string": "00:00:00;00",
      "value": "0"
    },
    "firstub": "0",
    "frames": {
      "string": "00:00:00;00",
      "value": "0"
    },
    "index": "0",
    "metadata": "",
    "metatcpid": "4294967294",
    "mp4proxydatarate": "2000000",
    "mp4proxyfcc": "1635148593",
    "mp4proxymode": "0",
    "mp4proxyscaledown": "2",
    "ms": "0",
    "name": "HBR-Live",
    "port": "5004",
    "previewenabled": "0",
    "protocol": "RTP",
    "rectc": "1",
    "starttc": "-1",
    "state": {
      "desc": "Not Running",
      "value": "8"
    },
    "streamaddress": "239.254.30.30",
    "streamduration": "0",
    "streamenabled": "1",
    "streamfile": "na",
    "streamport": "5004",
    "tc": {
      "string": "00:00:00;00",
      "value": "0"
```

```
   <channel index="1">
                                    . . .
   </channel>
  </group>
  <group index="1" key="DemoStream"
version="v5.1.0.13" autostart="1">
   <channel index="0">
    <name>Channel0</name>
    <capturefile/>
    <enabled>0</enabled>
    <address>239.0.0.0</address>
    <uuid/>
    <port>5004</port>
    <protocol>RTP</protocol>
    <rectc>0</rectc>
    <directory>
      D:\drastic.trunk\applications\web\netx\video\
streamTS
    </directory>
    <type>.ts</type>
    <capturecc>-1</capturecc>

<streamaddress>239.254.30.30</streamaddress
>
    <streamport>5004</streamport>
    <streamfile>na</streamfile>
    <streamenabled>1</streamenabled>
    <streamduration>0</streamduration>
    <previewenabled>0</previewenabled>
    <mp4proxymode>0</mp4proxymode>
    <mp4proxyfcc>1635148593</mp4proxyfcc>

<mp4proxydatarate>2000000</mp4proxydatarate
>

<mp4proxyscaledown>2</mp4proxyscaledown>
    <ms>0</ms>
    <bytes>0</bytes>
    <state desc="Not Running">8</state>
    <metadata/>
    <tc string="00:00:00;00">0</tc>
    <metatcpid>4294967294</metatcpid>
```

```
  },
  "tctype": {
    "string": "29.97",
    "value": "4"
  },
  "timecode": {
    "string": "00:00:00;00",
    "value": "0"
  },
  "type": ".mxf-op1a",
  "userbits": "0",
  "uuid": "",
  "vitctimecode": "3452816845",
  "vitcub": "3452816845"
 },
 "channel1": {
                . . .
 },
 "channel2": {
                . . .
 },
 "channel3": {
                . . .
 },
 "channel4": {
                . . .
 },
 "channel5": {
                . . .
 },
 "channel6": {
                . . .
 },
 "channel7": {
                . . .
 },
 "channel8": {
                . . .
 },
 "channel9": {
                . . .
 },
```

```
    <bcdtc/>
    <timecode
string="00:00:00;00">0</timecode>
    <firsttc string="00:00:00;00">0</firsttc>
    <frames string="00:00:00;00">0</frames>
    <firstub>0</firstub>
    <tctype string="29.97">4</tctype>
    <bcdub/>
    <userbits>0</userbits>
    <starttc>-1</starttc>
    <endtc>-1</endtc>
    <vitctimecode>3452816845</vitctimecode>
    <vitcub>3452816845</vitcub>
   </channel>
                                  . . .
  </group>
  <group index="2" key="DemoRestream"
version="0" autostart="1">
                                  . . .
  </group>
  <result>0</result>
  <resultstring>group returned</resultstring>
</client>
```

```
  "index": "0",
  "key": "DemoCapture",
  "version": "v5.1.0.13"
 },
 "group1": {
  "autostart": "1",
                    . . .
  "index": "1",
  "key": "DemoStream",
  "version": "v5.1.0.13"
 },
 "group2": {
                    . . .
 },
 "groups": "3",
 "netxbaseip": "192.168.50.100",
 "result": "0",
 "resultstring": "group returned",
 "version": "v7.1.0.3"
}
```

Or a single setting from a single group and all channels

http://127.0.0.1:1080/netxjson?request=get&client=0&group=1&channel=all&value=address

```
<client netxbaseip="192.168.50.100"
address="192.168.100.229" version="v7.1.0.1"
groups="3" autostart="0">
  <group index="1" key="DemoStream"
version="v5.1.0.13" autostart="1">
   <channel index="0">
    <address>239.0.0.0</address>
   </channel>
   <channel index="1">
    <address>239.0.0.0</address>
   </channel>
   <channel index="2">
    <address>239.0.0.0</address>
   </channel>
   <channel index="3">
```

```
{
 "address": "192.168.100.229",
 "autostart": "0",
 "group1": {
  "autostart": "1",
  "channel0": {
   "address": "239.0.0.0",
   "index": "0"
  },
  "channel1": {
   "address": "239.0.0.0",
   "index": "1"
  },
  "channel2": {
   "address": "239.0.0.0",
```

```
    <address>239.0.0.0</address>
   </channel>
   <channel index="4">
    <address>239.0.0.0</address>
   </channel>
   <channel index="5">
    <address>239.0.0.0</address>
   </channel>
   <channel index="6">
    <address>239.0.0.0</address>
   </channel>
   <channel index="7">
    <address>239.0.0.0</address>
   </channel>
   <channel index="8">
    <address>239.0.0.0</address>
   </channel>
   <channel index="9">
    <address>239.0.0.0</address>
   </channel>
  </group>
  <result>0</result>
  <resultstring>group returned</resultstring>
</client>
```

```
    "index": "2"
   },
   "channel3": {
    "address": "239.0.0.0",
    "index": "3"
   },
   "channel4": {
    "address": "239.0.0.0",
    "index": "4"
   },
   "channel5": {
    "address": "239.0.0.0",
    "index": "5"
   },
   "channel6": {
    "address": "239.0.0.0",
    "index": "6"
   },
   "channel7": {
    "address": "239.0.0.0",
    "index": "7"
   },
   "channel8": {
    "address": "239.0.0.0",
    "index": "8"
   },
   "channel9": {
    "address": "239.0.0.0",
    "index": "9"
   },
   "index": "1",
   "key": "DemoStream",
   "version": "v5.1.0.13"
  },
  "groups": "3",
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "group returned",
  "version": "v7.1.0.3"
}
```

3.1.1 previewenabled

Determine if preview JPEG capture is currently enabled

http://127.0.0.1:1080/netx?request=get&client=0&group=0&channel=0&value=previewenabled

<client netxbaseip="192.168.50.100"
address="192.168.100.229" version="v7.1.0.1"
groups="3" autostart="0">
  <group index="0" key="DemoCapture"
version="5.0.0.252" autostart="1">
    <channel index="0">
      <unknown_request/>
    </channel>
  </group>
  <result>0</result>
  <resultstring>group returned</resultstring>
</client>

{
  "address": "192.168.100.229",
  "autostart": "0",
  "group0": {
    "autostart": "1",
    "channel0": {
      "index": "0",
      "unknown_request": ""
    },
    "index": "0",
    "key": "DemoCapture",
    "version": "v5.0.0.252 "
  },
  "groups": "3",
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "group returned",
  "version": "v7.1.0.3"
}

3.1.2 preview

Getting a preview JPEG embedded in the XML response

http://127.0.0.1:1080/netx?request=get&client=0&group=0&channel=0&value=preview

<client netxbaseip="192.168.50.100"
address="192.168.100.229" version="v7.1.0.1"
groups="3" autostart="0">
  <group index="0" key="DemoCapture"
version="0" autostart="1">
    <channel index="0">
      <preview timestamp="95479532">

/9j/4AAQSkZJRgABAQEAYABgAAD//gAW9k= . . .

```
    </preview>
   </channel>
  </group>
  <result>0</result>
  <resultstring>group returned</resultstring>
</client>
```

The data returned is base 64 encoded jpeg data located in the following xml structure:

3.1.3 pts

Getting the current pts (presentation time stamp) values for audio, video and time code

http://127.0.0.1:1080/netx?request=get&client=0&group=0&channel=0&value=pts

```
<client netxbaseip="192.168.50.100"
address="192.168.100.229" version="v7.1.0.1"
groups="3" autostart="0">
  <group index="0" key="DemoCapture"
version="0" autostart="1">
    <channel index="0">
      <video_pts>23470552</video_pts>
      <audio_pts>23385674</audio_pts>
      <tc_pts>0</tc_pts>
      <days>0</days>
      <months>23475929</months>
      <years>24</years>
    </channel>
  </group>
  <result>0</result>
  <resultstring>group returned</resultstring>
</client>
```

```
{
  "address": "192.168.100.229",
  "autostart": "0",
  "group0": {
    "autostart": "1",
    "channel0": {
      "audio_pts": "0",
      "days": "0",
      "index": "0",
      "months": "6400221249",
      "tc_pts": "0",
      "video_pts": "18174762",
      "years": "5"
    },
    "index": "0",
    "key": "DemoCapture",
    "version": "v5.1.0.13"
  },
  "groups": "3",
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "group returned",
  "version": "v7.1.0.3"
}
```

3.1.4 datarate

Getting the last known stream data rate. By default, all rate types are returned.

http://127.0.0.1:1080/netx?request=get&client=0&group=0&channel=0&value=datarate

<client netxbaseip="192.168.50.100" address="192.168.100.229" version="v7.1.0.1" groups="3" autostart="0">
  <group index="0" key="DemoCapture" version="0" autostart="1">
    <channel index="0">
      <datarate_total timestamp="3452816845"/>
      <datarate_video timestamp="3452816845"/>
      <datarate_audio timestamp="3452816845"/>
    </channel>
  </group>
  <result>0</result>
  <resultstring>group returned</resultstring>
</client>


```
{
  "address": "192.168.100.229",
  "autostart": "0",
  "group0": {
    "autostart": "1",
    "channel0": {
      "datarate_audio": "",
      "datarate_total": "",
      "datarate_video": "",
      "index": "0"
    },
    "index": "0",
    "key": "DemoCapture",
    "version": "v5.1.0.13"
  },
  "groups": "3",
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "group returned",
  "version": "v7.1.0.3"
}
```


The value returned will contain rates for video/audio & total. Specific values can be retrieved by appending the desired type to the call:

http://127.0.0.1:1080/netx?request=get&client=0&group=0&channel=0&value=datarate&total=

<client netxbaseip="192.168.50.100" address="192.168.100.229" version="v7.1.0.1" groups="3" autostart="0">
  <group index="0" key="DemoCapture" version="0" autostart="1">
    <channel index="0">
      <datarate_total timestamp="3452816845"/>
    </channel>
  </group>
  <result>0</result>


```
{
  "address": "192.168.100.229",
  "autostart": "0",
  "group0": {
    "autostart": "1",
    "channel0": {
      "datarate_total": "",
      "index": "0"
    },
    "index": "0",
```

<resultstring>group returned</resultstring> </client>	"key": "DemoCapture", "version": "v5.1.0.13" }, "groups": "3", "netxbaseip": "192.168.50.100", "result": "0", "resultstring": "group returned", "version": "v7.1.0.3" }

http://127.0.0.1:1080/netxjson?request=get&client=0&group=0&channel=0&value=datarate&video=

<client netxbaseip="192.168.50.100"
address="192.168.100.229" version="v7.1.0.1"
groups="3" autostart="0">
<group index="0" key="DemoCapture"
version="0" autostart="1">
<channel index="0">
<datarate_video timestamp="3452816845"/>
</channel>
</group>
<result>0</result>
<resultstring>group returned</resultstring>
</client>
	{
	"address": "192.168.100.229",
	"autostart": "0",
	"group0": {
	"autostart": "1",
	"channel0": {
	"datarate_video": "",
	"index": "0"
	},
	"index": "0",
	"key": "DemoCapture",
	"version": "v5.1.0.13"
	},
	"groups": "3",
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "group returned",
	"version": "v7.1.0.3"
	}

3.1.5 property

Getting a stream property

http://127.0.0.1:1080/netx?request=get&client=0&group=0&channel=0&value=property&video_width=

<client netxbaseip="192.168.50.100" address="192.168.100.229" version="v7.1.0.1"	{ "address": "192.168.100.229",

```
groups="3" autostart="0">
  <group index="0" key="DemoCapture"
version="0" autostart="1">
    <channel index="0">
      <video_width
changed="0">1280</video_width>
    </channel>
  </group>
  <result>0</result>
  <resultstring>group returned</resultstring>
</client>
```

```
  "autostart": "0",
  "group0": {
    "autostart": "1",
    "channel0": {
      "index": "0",
      "video_width": {
        "changed": "0",
        "value": "1280"
      }
    },
    "index": "0",
    "key": "DemoCapture",
    "version": "v5.1.0.13"
  },
  "groups": "3",
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "group returned",
  "version": "v7.1.0.3"
}
```

The value returned in this case will be the width in pixels of the captured stream (see response below). An XML attribute is provided with each property to denote if there has been a change for some reason. Properties will not be available until the stream starts capturing. All properties can be retrieved at once by calling:

```
http://127.0.0.1:1080/netx?request=get&client=0&group=0&channel=0&value=property&all=
```

```
<client netxbaseip="192.168.50.100"
address="192.168.100.229" version="v7.1.0.1"
groups="3" autostart="0">
  <group index="0" key="DemoCapture"
version="0" autostart="1">
    <channel index="0">
      <previewenabled>1</previewenabled>
      <video_channel
changed="0">1</video_channel>
      <video_codec changed="0">0</video_codec>
      <video_bitdepth
changed="0">24</video_bitdepth>
      <video_rate changed="0">59.94</video_rate>
      <video_width
changed="0">1280</video_width>
```

```
{
  "address": "192.168.100.229",
  "autostart": "0",
  "group0": {
    "autostart": "1",
    "channel0": {
      "audio_bitdepth": {
        "changed": "0",
        "value": "16"
      },
      "audio_channel": {
        "changed": "0",
        "value": "0"
      },
      "audio_rate": {
```

<table>
<tr>
<td>

```
    <video_height
changed="0">720</video_height>
    <audio_channel
changed="0">2</audio_channel>
    <audio_bitdepth
changed="0">24</audio_bitdepth>
    <audio_rate
changed="0">48000</audio_rate>
   </channel>
  </group>
  <result>0</result>
  <resultstring>group returned</resultstring>
</client>
```

</td>
<td>

```
      "changed": "0",
      "value": "48000"
    },
    "index": "0",
    "previewenabled": "1",
    "video_bitdepth": {
      "changed": "0",
      "value": "24"
    },
    "video_channel": {
      "changed": "0",
      "value": "1"
    },
    "video_codec": {
      "changed": "1",
      "value": "1635148593"
    },
    "video_height": {
      "changed": "0",
      "value": "720"
    },
    "video_rate": {
      "changed": "0",
      "value": "59.94"
    },
    "video_width": {
      "changed": "0",
      "value": "1280"
    }
   },
   "index": "0",
   "key": "DemoCapture",
   "version": "v5.1.0.13"
  },
  "groups": "3",
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "group returned",
  "version": "v7.1.0.3"
}
```

</td>
</tr>
</table>

3.1.6 status

Getting a status list

http://127.0.0.1:1080/netxjson?request=get&client=0&group=0&channel=0&value=status

<client netxbaseip="192.168.50.100" address="192.168.100.229" version="v7.1.0.1" groups="3" autostart="0">
  <group index="0" key="DemoCapture" version="0" autostart="1">
    <channel index="0">
      <status count="1">
        <status0 index="0" timestamp="95468907" desc="meta tc pid changed:old">2</status0>
      </status>
    </channel>
  </group>
  <result>0</result>
  <resultstring>group returned</resultstring>
</client>


```
{
  "address": "192.168.100.229",
  "autostart": "0",
  "group0": {
    "autostart": "1",
    "channel0": {
      "index": "0",
      "status": {
        "count": "292",
        }
      }
    },
    "index": "0",
    "key": "DemoCapture",
    "version": "v5.1.0.13"
  },
  "groups": "3",
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "group returned",
  "version": "v7.1.0.3"
}
```


The XML returned will contain a list of status messages for the given channel(s). Provided is a status count and an index for each status message. The character value of each message specifies the severity level. All messages will be removed from the internal list once they have been retrieved. Any non-critical messages will be removed after 5 minutes.

0 = Information, 1 = Warning, 2 = Critical

<?xml version="1.0" encoding="UTF-8"?>
<client address="192.168.100.176" version=v4.2.0.274 groups="1" autostart=”1”>
  <group index="1" key="GroupName" autostart=”1”>
    <channel index="0">
      <status count=1”>
        <status0 index=”0” timestamp=”6516120” desc=”meta tc pid changed”>2</status0>

```
        </status>
      </channel>
    </group>
  <result>0</result>
  <resultstring>group returned</resultstring>
</request>
```

3.1.7 streamstate

Get the running state of a capturing stream

http://192.168.100.229:1080/netx?request=get&client=0&group=0&channel=0&value=streamstate

<client netxbaseip="192.168.50.100" address="192.168.100.229" version="v7.1.0.1" groups="3" autostart="0">
  <group index="0" key="DemoCapture" version="0" autostart="1">
    <channel index="0">
      <streamstate>requested</streamstate>
    </channel>
  </group>
  <result>0</result>
  <resultstring>success</resultstring>
</client>


```
{
  "address": "192.168.100.229",
  "autostart": "0",
  "group0": {
    "autostart": "1",
    "channel0": {
      "index": "0",
      "streamstate": "requested"
    },
    "index": "0",
    "key": "DemoCapture",
    "version": "v5.1.0.13"
  },
  "groups": "3",
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "group returned",
  "version": "v7.1.0.3"
}
```


This command retrieves the running information on a given stream. This includes information derived from the incoming stream like video width/height, audio frequency, packets captured, pids and the UUID of the stream. This command is slightly different than the other gets, as it requires two calls to get the information. The first sets up the request from Net-X-Base to Net-X-Code, and the second actually gets the data. The first call will return:

<client address="192.168.100.176" version="v5.0.0.39" groups="1" autostart=”1”>

```
  <group index="0" key="Default" version="v5.0.0.39" autostart="1">
    <channel index="0">
      <streamstate>requested</streamstate>
    </channel>
  </group>
  <result>0</result>
  <resultstring>success</resultstring>
</client>
```

The second will then return with the information:

```
<client address="192.168.100.176" version="v5.0.0.39" groups="1" autostart="1">
  <result>0</result>
  <resultstring>success</resultstring>
  <group index="0" key="Default" version="v5.0.0.39" autostart="1">
    <channel index="0">
      <streamstate>
        <netxcode-stream id="0">
          <video_width>1280</video_width>
          <video_height>720</video_height>
          <video_bits>24</video_bits>
          <video_rate>60000</video_rate>
          <video_scale>1001</video_scale>
          <audio_frequency>48000</audio_frequency>
          <audio_bits>16</audio_bits>
          <uuid>00000000000000000000000000000000</uuid>
          <framecount>423686</framecount>
          <length>557570776</length>
          <packets_total>2965789</packets_total>
          <packets_video>2282451</packets_video>
          <packets_audio>64638</packets_audio>
          <packets_tc>10275</packets_tc>
          <packets_metadata>0</packets_metadata>
          <packets_cc>0</packets_cc>
          <pid_video>2001</pid_video>
          <pid_audio>3001</pid_audio>
          <pid_audio1>4294967294</pid_audio1>
          <pid_audio2>4294967294</pid_audio2>
          <pid_audio3>4294967294</pid_audio3>
          <pid_dolby>4294967294</pid_dolby>
          <pid_tc>501</pid_tc>
          <pid_cc>4294967294</pid_cc>
        </netxcode-stream>
```

```
    </streamstate>
   </channel>
  </group>
</client>
```

3.1.8 state

Return the Net-X-Base status including last command, time of last command and times of the last communication with a Net-X-Cmd.
NOTE: Possible JSON issue – please check

http://127.0.0.1:1080/netxjson?request=get&netxbase=state


```
<netxbase>
  <result>0</result>
  <resultstring>output state
returned</resultstring>
  <request-ms>96225028</request-ms>
  <client-info>
    <state>0</state>
    <ms>96224882</ms>
    <data>
group=0&channel=0&stats=&metatcpid=501&uui
d=55E869401C2FE911947D15250A4E0C00&tc=
4021154&timecode=4021159&tctype=32&bcdub=
40608010&userbits=276848704&lastcapturems=
299122&lastcapturebytes=2964720304&vitctimec
ode=2942239&vitcub=276848704&firsttc=401181
4&firstub=276848704&frames=17905&starttc=-
1&endtc=-1
    </data>
    <group-info>
      <state>0</state>
      <ms>96224882</ms>
      <channel-info>
        <state>0</state>
        <ms>96224882</ms>
      </channel-info>
    </group-info>
  </client-info>
  <client-command>
    <state>0</state>
```

```
{
  "client-command": {
    "data":
"address=192.168.100.229&group=0&channel=0
&param=preview&value=1",
    "ms": "165793269",
    "state": "0"
  },
  "client-handler": {
    "ms": "166267733",
    "state": "0"
  },
  "client-info": {
    "data":
"group=0&channel=3&pts=&video_pts=63784083
66&audio_pts_0=6378434904&audio_pts_1=0&a
udio_pts_2=0&audio_pts_3=0&audio_pts_4=0&a
udio_pts_5=0&audio_pts_6=0&audio_pts_7=0&tc
_pts=6378468426&days=5&months=4",
    "group-info": {
      "channel-info": {
        "ms": "166267736",
        "state": "0"
      },
      "ms": "166267736",
      "state": "0"
    },
    "ms": "166267736",
    "state": "0"
```

<ms>0</ms>
<data/>
</client-command>
<start-command>
<state>0</state>
<ms>0</ms>
</start-command>
<stop-command>
<state>0</state>
<ms>0</ms>
</stop-command>
<client-read>
<state>0</state>
<ms>96224881</ms>
</client-read>
<client-handler>
<state>0</state>
<ms>96224881</ms>
</client-handler>
<client-command>
<state>0</state>
<ms>96100291</ms>
</client-command>
</netxbase>
	},
	"client-read": {
	"ms": "166267733",
	"state": "0"
	},
	"request-ms": "166267973",
	"result": "0",
	"resultstring": "output state returned",
	"start-command": {
	"ms": "0",
	"state": "0"
	},
	"stop-command": {
	"ms": "0",
	"state": "0"
	}
	}

3.1.9 xmldisable

Disable XML sidecar generation for captured files

This command gets the state of XML sidecar generation

http://127.0.0.1/netx?request=get&client=0&group=0&channel=0&value=xmldisable

<client netxbaseip="192.168.50.100"
address="192.168.100.229" version="v7.1.0.1"
groups="3" autostart="0">
<group index="0" key="DemoCapture"
version="0" autostart="1">
<channel index="0">
<xmldisable>0</xmldisable>
</channel>
	{
	"address": "192.168.100.229",
	"autostart": "0",
	"group0": {
	"autostart": "1",
	"channel0": {
	"index": "0",
	"xmldisable": "0"

</group> <result>0</result> <resultstring>group returned</resultstring> </client>	}, "index": "0", "key": "DemoCapture", "version": "v5.1.0.13" }, "groups": "3", "netxbaseip": "192.168.50.100", "result": "0", "resultstring": "group returned", "version": "v7.1.0.3" }

If set to 0, no sidecar XML files will be generated. If set to one, standard Drastic XML sidecar files will be generated.

3.1.10 autostart

Autostart is not a command, like the others in this list. It is included in all returns as an attribute of the <client> and <group> tags. Autostart, if enabled, causes any stream that is lost due to Net-X-Cmd or Net-X-Code stopping, to be automatically restarted when they are restarted. There are two levels of autostart:

1. <client> autostart must be enabled for any kind of autostart to occur on a client (Net-X-Cmd) server. If this is disabled, the next level is ignored.
2. <group> if the client autostart is enabled, then the Net-X-Cmd will look at the group autostarts to determine if a group should be restarted when Net-X-Cmd or Net-X-Code gets closed. This allows the caller to set up a server that only restarts some channels automatically.

3.1.11 getDiscontinuities

Get the XML containing the info on discontinuities encountered during a capture.

You must make this request in 2 parts; first send a set request with the filename of the source, ex:

```
http://127.0.0.1:1080/netx?request=set&client=0&command=getDiscontinuities&source=M:\t3media\Streams\espn\17-06-13_Discontinuities\Channel0.mp4
```

<request netxbaseip="192.168.50.100" type="command=getDiscontinuities">
<getDiscontinuities success="1">success</getDiscontinuities>
	{
	"getDiscontinuities": {
	"success": "1",
	"value": "success"

<result>0</result> <resultstring>success</resultstring> </request>	}, "netxbaseip": "192.168.50.100", "result": "0", "resultstring": "success", "type": "command=getDiscontinuities" }

Then send the get request with the same source, ex:

http://127.0.0.1:1080/netx?request=get&client=0&command=getDiscontinuities&source=M:\t3media\Streams\espn\17-06-13_Discontinuities\Channel0.mp4

<getdiscontinuities> <result>0</result> <resultstring>discontinuities returned</resultstring> <DISCONTINUITIES> <Discontinuity type="video" duration="14231ms" time="11:52:02 Wednesday, June 14, 2017" frame="1210" LastITimeCode="4094240" DisconTimeCode="4094250" timecodetype="d">18:58:25;50</Discontinuity> <Discontinuity type="audio" duration="14308ms" time="11:52:02 Wednesday, June 14, 2017" sample="968967" LastITimeCode="4094240" DisconTimeCode="4094250" timecodetype="d">18:58:25;50</Discontinuity> <Discontinuity type="timecode" time="11:52:02 Wednesday, June 14, 2017" frame="1212" LastITimeCode="4094240" DisconTimeCode="4094252" timecodetype="d" comment="CurrentTimeCode: 4095050">18:58:25;52</Discontinuity> <Discontinuity type="video" duration="6641ms" time="11:52:25 Wednesday, June 14, 2017" frame="2241" LastITimeCode="4096070" DisconTimeCode="4096078" timecodetype="d">18:58:56;18</Discontinuity> </DISCONTINUITIES> </getdiscontinuities>	**NOTE: Currently JSON returns are not supported for this command**

3.1.12 getCopyInOut

Get the byte ranges necessary to clip a file in a given time code range. This can be made via the REST API or the command line.

For the REST API, you must make this request in 2 parts; first send a set request with the filename of the source, an in time code, an out time code, temp folder, and optionally a byte alignment. If not supplied the alignment will be 1 byte.

Request ex:

```
http://127.0.0.1:1080/netx?request=set&client=0&command=getCopyInOut&source=M:\
t3media\Streams\NBA\NBA_Reference_Test_Files\
amberfin_new_0000750142.rtin&in=22:29:28;14&out=22:29:38;14&alignment=4096
```

```
<request netxbaseip="192.168.50.100"
type="command=getCopyInOut">
  <getCopyInOut
success="1">success</getCopyInOut>
  <result>0</result>
  <resultstring>success</resultstring>
</request>
```

```
{
  "getCopyInOut": {
    "success": "1",
    "value": "success"
  },
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "type": "command=getCopyInOut"
}
```

And you should get a response saying the command was understood, and is processing.

Then send the exact same command, except replace the ‘set’ with a ‘get’ to get the response to the previous request.

Get request response ex:

```
http://127.0.0.1:1080/netx?request=get&client=0&command=getCopyInOut&source=M:\
t3media\Streams\NBA\NBA_Reference_Test_Files\
amberfin_new_0000750142.rtin&in=22:29:28;14&out=22:29:38;14&alignment=4096
```

```
<getcopyinout>
  <result>0</result>
  <resultstring>returning copy in out</resultstring>
  <copyinoutinfo>
    <originalFile>
```

```
{
  "copyinoutinfo": {
    "in": "147894272",
    "originalFile": "\\\\192.168.101.200\\media\\t3media\\
Streams\\NBA\\NBA_Reference_Test_Files\\
```

\\192.168.101.200\media\t3media\Streams\NBA\ NBA_Reference_Test_Files\ amberfin_new_0000750142.mxf </originalFile> <tempFile>amberfin_new_0000750142_147894272_298274816.mxf</tempFile> <in>147894272</in> <out>298274816</out> </copyinoutinfo> </getcopyinout>	amberfin_new_0000750142.mxf", "out": "298274816", "tempFile": "amberfin_new_0000750142_147894272_298274816.mxf" }, "result": "0", "resultstring": "returning copy in out" }

And you will get back an XML with the source filename, temp name to copy the segment to, and the uint64 start and end point to copy from.

The returned XML will contain the file you need to get off tape (originalFile) as well as a target filename (tempFile). You'll need to bring the byte ranges (in, out) of the originalFiles off the tape and name according to their tempFile. Put all the temp files in the same folder (<tempFolder>). Then make the copy request as normal with an additional parameter: &tempfolder=<tempFolder>. See “Command – Copy, Convert and PFR” below for info on copy requests.

Restore using the temp files ex:

http://127.0.0.1:1080/netx?request=set&client=&command=copy&profile=wrap&source=M:\t3media\Streams\NBA\NBA_Reference_Test_Files\amberfin_new_0000750142.rtin&in=22:29:28;14&out=22:29:38;14&alignment=4096&target=m:\restorepoint\partial\amberfin_clip.mxf&tempfolder=D:\Record

<request netxbaseip="192.168.50.100"
type="command=copy">
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "success",
	"type": "command=copy"
	}

You may get multiple files back in the XML, this means the audio and video are in separate files. This will happen if the source file is a Panasonic P2, AS-02, DCP or other multi part file. In this case, copy the data from each source file to its temp file in the temp folder, then pass the temp folder to the translate as above.

Return ex:
<getCopyInOut>

```
    <copyInOutInfo>
        <originalFile>
            C:\Users\Ryan\Videos\record0\Channel0\media\Channel0_v00.mxf
        </originalFile>
        <tempFile>Channel0_v00_87706372_191637286.mxf</tempFile>
        <in>87706372</in>
        <out>191637286</out>
    </copyInOutInfo>
    <copyInOutInfo>
        <originalFile>
            C:\Users\Ryan\Videos\record0\Channel0\media\Channel0_a00.mxf
        </originalFile>
        <tempFile>Channel0_a00_2526245_5519636.mxf</tempFile>
        <in>2526245</in>
        <out>5519636</out>
    </copyInOutInfo>
    .
    .
    .
</getCopyInOut>
```

3.1.12.1 Tape PFR using the command line

The getCopyInOut command for partial file restore from binary object are also available via Net-X-Copy's command line. To get the in/out positions in the source file, send a getCopyInOut command to Net-X-Copy

NetXCopy -p getCopyInOut -s M:\t3media\Streams\NBA\NBA_Reference_Test_Files\ amberfin_new_0000750142.rtin -in 22:29:28;14 -out 22:29:38;14 -alignment 4096

This will return the source file(s) and the start and end position for the copy
source=M:\t3media\Streams\NBA\NBA_Reference_Test_Files\
amberfin_new_0000750142.rtin&in=22:29:28;14&out=22:29:38;14&alignment=4096&getCopyInOut=&
originalFile=\\192.168.101.200\media\t3media\Streams\NBA\NBA_Reference_Test_Files\
amberfin_new_0000750142.mxf&tempFile=amberfin_new_0000750142_147894272_298274816.mxf
&u64in=147894272&u64out=298274816

Once you have copied the area from the 'originalFile=' from 'u64in=' to 'u64out=' into a temporary directory, you can then call the partial file restore as you normally would, but with a 'tempfolder=' set pointing to where the binary object was restored to:

NetXCopy -p wrap -s M:\t3media\Streams\NBA\NBA_Reference_Test_Files

amberfin_new_0000750142.rtin -in 22:29:28;14 -out 22:29:38;14 -alignment 4096 -t m:\restorepoint\partial\amberfin_clip.mxf -tempfolder d:\record

3.1.13 getFirstAndLastTimecode

Get the first and last time code locations from a file.

You must make this request in 2 parts, first send a set request with the filename of the source, ex:

```
http://127.0.0.1:1080/netx?request=set&client=0&command=getFirstAndLastTimecode&source=M:\t3media\Streams\NBA\NBA_Reference_Test_Files\amberfin_0000683858.rtin
```

<request netxbaseip="192.168.50.100" type="command=getFirstAndLastTimecode">
  <getFirstAndLastTimecode success="1">success</getFirstAndLastTimecode>
  <result>0</result>
  <resultstring>success</resultstring>
</request>

```
{
  "getFirstAndLastTimecode": {
    "success": "1",
    "value": "success"
  },
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "type": "command=getFirstAndLastTimecode"
}
```

Then send the get request with the same source, ex:

```
http://127.0.0.1:1080/netx?request=get&client=0&command=getFirstAndLastTimecode&source=M:\t3media\Streams\NBA\NBA_Reference_Test_Files\amberfin_0000683858.rtin
```

<getfirstandlasttimecode>
  <source>
    M:\t3media\Streams\NBA\NBA_Reference_Test_Files\amberfin_0000683858.rtin
  </source>
  <tcIn>19:45:01;01</tcIn>
  <tcOut>22:05:16;11</tcOut>
  <result>0</result>
  <resultstring>returning first and last timecode</resultstring>
</getfirstandlasttimecode>

```
{
  "result": "0",
  "resultstring": "returning first and last timecode",
  "source": "M:\\t3media\\Streams\\NBA\\NBA_Reference_Test_Files\\amberfin_0000683858.rtin",
  "tcIn": "19:45:01;01",
  "tcOut": "22:05:16;11"
}
```

3.2 Command – Copy, Convert and PFR

UPDATE: Copy status and purge calls now require the group id assigned by the system. It starts at -2 and counts down into negative numbers. -1 will apply the command to all copy groups for the given client.

COMMANDS: Require client address only. The Net-X-Copy application will only run as a single instance on a given machine. Group id and channel id will be ignored. The command will always equal copy, regardless of whether the desired effect is copy or convert.

PARAMETERS: source, target, ackfile, profile, type, tcin, tcout, absin, absout, tcoffset, tc, ub, width, height, copy, dest, ccfile, afile, vfile, stereo, aroute, uuid, kilobitrate, h26xprofile, h26xlevel, encodemode, gopsize, tempfile, alignment, flags, metadata, fg, fc, pisrc, pidst, pisize, piframe, piskip,

TRIGGERS: clear, abort, purge

3.2.1 set – initiate copy/convert/pfr

Can be used to initiate a copy or conversion, if the profile parameter is not specified, then a simple file copy will be performed. Additionally, the set command can be used to clean up old or stale copy requests, or cancel/clear active and/or pending copy requests. The major types of copy include:

- wrap – re wrap the audio/video essence without recompressing
- index – make an RTIN index of a file for later partial file restore
- getCopyInOut – find a byte range to restore from tape for a PFR
- a series of conversion types (mxf-xdcam, mov-proreshq, etc). These will re compress the created clip to a certain video/audio compression set in a particular container. For a complete list, please see the profile list below
- When converting, there is also a type parameter. This lets you override the original type, in the case of a wrap, or the set type, in the case of a converting profile, to some standard containers. Please note, not all containers support all codecs.

A typical copy/conversion initial call would be as follows:

```
http://127.0.0.1:1080/netx?request=set&client=0&command=copy&profile=copy&source=D:/record/sourcefile.mp4&target= D:/record/targetfile.mp4
```

<request netxbaseip="192.168.50.100"	{

type="command=copy"> <result>0</result> <resultstring>success</resultstring> </request>	"netxbaseip": "192.168.50.100", "result": "0", "resultstring": "success", "type": "command=copy" }

There is also a multi source/target version of the copy. The profile, in and out must be the same, but multiple source and associated targets can be sent as one command.

http://127.0.0.1:1080/netx?request=set&client=0&command=copy&source=d:/Record/file001.mov&target=D:/Record/target001.mov&source=D:/Record/file002.mov&target=D:/Record/target002.mov&profile=copy

<request netxbaseip="192.168.50.100"
type="command=copy">
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "success",
	"type": "command=copy"
	}

To create an RTIN index of a file, use the index command. If the target directory and file name are not specified, then the RTIN will use the same base directory and file name of the source file, with the .rtin extension. Please note, when restoring the index file, the source file must be in its original location or in the same directory as the index file, so that it can be found.

http://127.0.0.1:1080/netx?request=set&client=0&command=copy&profile=index&source=D:\record\amberfin_0000572516.mxf&target=D:\record\amberfin_0000572516.mxf.rtin

<request netxbaseip="192.168.50.100"
type="command=copy">
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "success",
	"type": "command=copy"
	}

To run the copy commands on a particular server, the client is normally specified as an IP address with 'client=0'. If you are running multiple servers, Net-X-Base can automatically select the next available server. To use this round robin method, specify 'client=any' in the command.

http://127.0.0.1:1080/netx?request=set&client=any&command=copy&profile=wrap&source=/mnt/server/media/source.mxf&target=/home/user/Videos/edit.mxf&tcin=01:01:30:00&tcout=01:02:00:00

<request netxbaseip="192.168.50.100"	{

type="command=copy"> <result>0</result> <resultstring>success</resultstring> </request>	"netxbaseip": "192.168.50.100", "result": "0", "resultstring": "success", "type": "command=copy" }

To create a picon (JPEG picture icon) of the frame at time code 01:01:30:00 at 10% of the original file, use a command similar to this:

```
http://127.0.0.1:1080/netx?request=set&client=any&command=copy&pisrc=D:\record\
amberfin_0000572516.mxf&pidst=D:\recored\
edit_01013000.picon.jpg&piSize=10&piframe=20:45:30:00
```

<request netxbaseip="192.168.50.100"
type="command=copy">
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "success",
	"type": "command=copy"
	}

If the profile is not specified and the tcin or tcout parameters are, a copy will take place instead of a conversion. Specifying the profile will initiate a conversion, if the tcin and tcout parameters are not specified then the entire file will be converted.

3.2.2 set/get – cardinfo (camera card clip lists)

To get the clips from a camera card, send a set to start the process.

http://127.0.0.1:1080/netxjson?request=set&client=any&command=copy&profile=cardinfo&source=M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\PX270MP_LasVegas\

<request netxbaseip="192.168.50.100" type="command=copy">
  <cardinfo success="1">0</cardinfo>
  <result>0</result>
  <resultstring>success</resultstring>
</request>

```
{
  "cardinfo": {
    "success": "1",
    "value": "0"
  },
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "type": "command=copy"
}
```

You will probably receive a few of these ‘fail, request already made’ returns until the cardinfo is ready. At that point, you will get back the actual cardinfo:

http://127.0.0.1:1080/netxjson?request=get&client=any&command=copy&profile=cardinfo&source=M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\PX270MP_LasVegas\

<request netxbaseip="192.168.50.100" type="command=copy">
  <cardinfo success="1">0</cardinfo>
  <result>0</result>
  <resultstring>success</resultstring>
</request>

<copy>
  <address>127.0.0.1</address>
  <netxbaseip>192.168.50.100</netxbaseip>
  <result>0</result>
  <resultstring>no data from previous metadata call found yet</resultstring>
</copy>

```
{
  "cardinfo": {
    "success": "1",
    "value": "0"
  },
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "no data from previous metadata call found yet",
  "type": "command=copy"
}
```

Once the set is sent, you can request the results by sending the same command with a get.

http://127.0.0.1:1080/netxjson?request=get&client=any&command=copy&profile=cardinfo&source=M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\PX270MP_LasVegas\


```
<copy>
  <address>192.168.100.229</address>
  <netxbaseip>192.168.50.100</netxbaseip>
  <result>0</result>
  <resultstring>Cardinfo returned</resultstring>
  <cardinfo source="M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\">
    <clip index="0">
      <filenamepath>
M:\t3media\Streams\MXF\Panasonic\Ultra-
LongGop25\PX270MP_LasVegas\CONTENTS\
AVCLIP\009RA3LL.MXF
      </filenamepath>
      <clipname>009RA3LL</clipname>
    </clip>
    <clip index="1">
      <filenamepath>
M:\t3media\Streams\MXF\Panasonic\Ultra-
LongGop25\PX270MP_LasVegas\CONTENTS\
AVCLIP\010UU0ZI.MXF
      </filenamepath>
      <clipname>010UU0ZI</clipname>
    </clip>
    <clip index="2">
      <filenamepath>
M:\t3media\Streams\MXF\Panasonic\Ultra-
LongGop25\PX270MP_LasVegas\CONTENTS\
AVCLIP\011IQ7HW.MXF
      </filenamepath>
      <clipname>011IQ7HW</clipname>
    </clip>
    <clip index="3">
      <filenamepath>
M:\t3media\Streams\MXF\Panasonic\Ultra-
LongGop25\PX270MP_LasVegas\CONTENTS\
AVCLIP\012OQ0MQ.MXF
      </filenamepath>
      <clipname>012OQ0MQ</clipname>
    </clip>
```

```
{
  "address": "192.168.100.229",
  "cardinfo": {
    "clip0": {
      "clipname": "009RA3LL",
      "filenamepath": "M:\\t3media\\Streams\\MXF\\
Panasonic\\Ultra-LongGop25\\
PX270MP_LasVegas\\CONTENTS\\AVCLIP\\
009RA3LL.MXF",
      "index": "0"
    },
    "clip1": {
      "clipname": "010UU0ZI",
      "filenamepath": "M:\\t3media\\Streams\\MXF\\
Panasonic\\Ultra-LongGop25\\
PX270MP_LasVegas\\CONTENTS\\AVCLIP\\
010UU0ZI.MXF",
      "index": "1"
    },
    "clip10": {
      "clipname": "020WO9TY",
      "filenamepath": "M:\\t3media\\Streams\\MXF\\
Panasonic\\Ultra-LongGop25\\
PX270MP_LasVegas\\CONTENTS\\AVCLIP\\
020WO9TY.MXF",
      "index": "10"
    },
    "clip11": {
      "clipname": "021XE8GX",
      "filenamepath": "M:\\t3media\\Streams\\MXF\\
Panasonic\\Ultra-LongGop25\\
PX270MP_LasVegas\\CONTENTS\\AVCLIP\\
021XE8GX.MXF",
      "index": "11"
    },
    "clip12": {
      "clipname": "022NP0YW",
      "filenamepath": "M:\\t3media\\Streams\\MXF\\
Panasonic\\Ultra-LongGop25\\
PX270MP_LasVegas\\CONTENTS\\AVCLIP\\
```

```
    <clip index="4">
      <filenamepath>
M:\t3media\Streams\MXF\Panasonic\Ultra-
LongGop25\PX270MP_LasVegas\CONTENTS\
AVCLIP\014MJ0XQ.MXF
      </filenamepath>
      <clipname>014MJ0XQ</clipname>
    </clip>
    <clip index="5">
      <filenamepath>
M:\t3media\Streams\MXF\Panasonic\Ultra-
LongGop25\PX270MP_LasVegas\CONTENTS\
AVCLIP\015KF5QO.MXF
      </filenamepath>
      <clipname>015KF5QO</clipname>
    </clip>
    <clip index="6">
      <filenamepath>
M:\t3media\Streams\MXF\Panasonic\Ultra-
LongGop25\PX270MP_LasVegas\CONTENTS\
AVCLIP\016SP1HV.MXF
      </filenamepath>
      <clipname>016SP1HV</clipname>
    </clip>
    <clip index="7">
      <filenamepath>
M:\t3media\Streams\MXF\Panasonic\Ultra-
LongGop25\PX270MP_LasVegas\CONTENTS\
AVCLIP\017UE4IK.MXF
      </filenamepath>
      <clipname>017UE4IK</clipname>
    </clip>
    <clip index="8">
      <filenamepath>
M:\t3media\Streams\MXF\Panasonic\Ultra-
LongGop25\PX270MP_LasVegas\CONTENTS\
AVCLIP\018WI7GJ.MXF
      </filenamepath>
      <clipname>018WI7GJ</clipname>
    </clip>
    <clip index="9">
      <filenamepath>
```

```
022NP0YW.MXF",
      "index": "12"
    },
    "clip2": {
      "clipname": "011IQ7HW",
      "filenamepath": "M:\\t3media\\Streams\\MXF\\
Panasonic\\Ultra-LongGop25\\
PX270MP_LasVegas\\CONTENTS\\AVCLIP\\
011IQ7HW.MXF",
      "index": "2"
    },
    "clip3": {
      "clipname": "012OQ0MQ",
      "filenamepath": "M:\\t3media\\Streams\\MXF\\
Panasonic\\Ultra-LongGop25\\
PX270MP_LasVegas\\CONTENTS\\AVCLIP\\
012OQ0MQ.MXF",
      "index": "3"
    },
    "clip4": {
      "clipname": "014MJ0XQ",
      "filenamepath": "M:\\t3media\\Streams\\MXF\\
Panasonic\\Ultra-LongGop25\\
PX270MP_LasVegas\\CONTENTS\\AVCLIP\\
014MJ0XQ.MXF",
      "index": "4"
    },
    "clip5": {
      "clipname": "015KF5QO",
      "filenamepath": "M:\\t3media\\Streams\\MXF\\
Panasonic\\Ultra-LongGop25\\
PX270MP_LasVegas\\CONTENTS\\AVCLIP\\
015KF5QO.MXF",
      "index": "5"
    },
    "clip6": {
      "clipname": "016SP1HV",
      "filenamepath": "M:\\t3media\\Streams\\MXF\\
Panasonic\\Ultra-LongGop25\\
PX270MP_LasVegas\\CONTENTS\\AVCLIP\\
016SP1HV.MXF",
      "index": "6"
```

<table>
<tr>
<td>M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\PX270MP_LasVegas\CONTENTS\AVCLIP\019GU1RS.MXF
</filenamepath>
<clipname>019GU1RS</clipname>
</clip>
<clip index="10">
<filenamepath>
M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\PX270MP_LasVegas\CONTENTS\AVCLIP\020WO9TY.MXF
</filenamepath>
<clipname>020WO9TY</clipname>
</clip>
<clip index="11">
<filenamepath>
M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\PX270MP_LasVegas\CONTENTS\AVCLIP\021XE8GX.MXF
</filenamepath>
<clipname>021XE8GX</clipname>
</clip>
<clip index="12">
<filenamepath>
M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\PX270MP_LasVegas\CONTENTS\AVCLIP\022NP0YW.MXF
</filenamepath>
<clipname>022NP0YW</clipname>
</clip>
</cardinfo>
</copy></td>
<td>},
"clip7": {
"clipname": "017UE4IK",
"filenamepath": "M:\\t3media\\Streams\\MXF\\Panasonic\\Ultra-LongGop25\\PX270MP_LasVegas\\CONTENTS\\AVCLIP\\017UE4IK.MXF",
"index": "7"
},
"clip8": {
"clipname": "018WI7GJ",
"filenamepath": "M:\\t3media\\Streams\\MXF\\Panasonic\\Ultra-LongGop25\\PX270MP_LasVegas\\CONTENTS\\AVCLIP\\018WI7GJ.MXF",
"index": "8"
},
"clip9": {
"clipname": "019GU1RS",
"filenamepath": "M:\\t3media\\Streams\\MXF\\Panasonic\\Ultra-LongGop25\\PX270MP_LasVegas\\CONTENTS\\AVCLIP\\019GU1RS.MXF",
"index": "9"
},
"source": "M:\\t3media\\Streams\\MXF\\Panasonic\\Ultra-LongGop25\\PX270MP_LasVegas\\"
},
"netxbaseip": "192.168.50.100",
"result": "0",
"resultstring": "Cardinfo returned"
}</td>
</tr>
</table>

3.2.3 set/get – clipfiles (files included in a clip)

To get the files from a clip, send a set to start the process.

```
http://127.0.0.1:1080/netxjson?
request=set&client=any&command=copy&profile=clipfiles&source=M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\PX270MP_LasVegas\Contents\AVCLIP\009RA3LL.MXF
```

<request netxbaseip="192.168.100.229"
type="command=copy">
  <clipfiles success="1">0</clipfiles>
  <result>0</result>
  <resultstring>success</resultstring>
</request>
	{
	  "clipfiles": {
	  "success": "1",
	  "value": "0"
	  },
	  "netxbaseip": "192.168.100.229",
	  "result": "0",
	  "resultstring": "success",
	  "type": "command=copy"
	}

You will probably receive a few of these ‘fail, request already made’ returns until the clipfiles is ready. At that point, you will get back the actual cardinfo:

http://127.0.0.1:1080/netxjson?request=get&client=any&command=copy&profile=clipfiles&source=M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\PX270MP_LasVegas\Contents\AVCLIP\009RA3LL.MXF

<copy>
  <address>127.0.0.1</address>
  <netxbaseip>192.168.50.100</netxbaseip>
  <result>0</result>
  <resultstring>no data from previous clipfiles call found yet</resultstring>
</copy>
	{
	  "address": "192.168.100.229",
	  "completed": "0",
	  "netxbaseip": "192.168.100.229",
	  "result": "0",
	  "resultstring": "no data from previous clipfiles call found yet"
	}

Once the set is sent, you can request the results by sending the same command with a get.

http://127.0.0.1:1080/netxjson?request=get&client=any&command=copy&profile=clipfiles&source=M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\PX270MP_LasVegas\Contents\AVCLIP\009RA3LL.MXF

<?xml version="1.0" encoding="UTF-8"?>
<copy>
  <address>192.168.100.229</address>
  <netxbaseip>192.168.100.229</netxbaseip>
  <completed>1</completed>
  <result>0</result>
  <resultstring>ClipFiles returned</resultstring>
  <clipfiles source="M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\PX270MP_LasVegas\Contents\AVCLIP\
	{
	  "address": "192.168.100.229",
	  "clipfiles": {
	  "info": "",
	  "info1": {
	  "channels": {
	  "location": "2",
	  "value": "1"
	  },
	  "file0": {

```
009RA3LL.MXF">
   <video index="0">
      <file index="0">
         <filename>M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\Contents\AVCLIP\
009RA3LL.MXF</filename>
            <length>483</length>
         </file>
      </video>
      <file index="0">
         <filename>M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\Contents\AVCLIP\
009RA3LL.MXF</filename>
            <length>483</length>
         </file>
         <channels location="4">1</channels>
         <file index="0">
         <filename>M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\Contents\AVCLIP\
009RA3LL.MXF</filename>
            <length>483</length>
         </file>
         <channels location="8">1</channels>
         <file index="0">
         <filename>M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\Contents\AVCLIP\
009RA3LL.MXF</filename>
            <length>483</length>
         </file>
         <channels location="16">1</channels>
         <file index="0">
         <filename>M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\Contents\AVCLIP\
009RA3LL.MXF</filename>
            <length>483</length>
         </file>
         <channels location="32">1</channels>
```

```
      "filename": "M:\\t3media\\Streams\\MXF\\
Panasonic\\Ultra-LongGop25\\
PX270MP_LasVegas\\Contents\\AVCLIP\\
009RA3LL.MXF",
      "index": "0",
      "length": "483"
    },
    "index": "1"
  },
  "source": "M:\\t3media\\Streams\\MXF\\
Panasonic\\Ultra-LongGop25\\
PX270MP_LasVegas\\Contents\\AVCLIP\\
009RA3LL.MXF",
  "video0": {
    "file0": {
      "filename": "M:\\t3media\\Streams\\MXF\\
Panasonic\\Ultra-LongGop25\\
PX270MP_LasVegas\\Contents\\AVCLIP\\
009RA3LL.MXF",
      "index": "0",
      "length": "483"
    },
    "index": "0"
  }
 },
 "completed": "1",
 "netxbaseip": "192.168.100.229",
 "result": "0",
 "resultstring": "ClipFiles returned"
}
```

```
        <file index="0">
        <filename>M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\Contents\AVCLIP\
009RA3LL.MXF</filename>
            <length>483</length>
        </file>
        <channels location="64">1</channels>
        <file index="0">
        <filename>M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\Contents\AVCLIP\
009RA3LL.MXF</filename>
            <length>483</length>
        </file>
        <channels location="128">1</channels>
        <file index="0">
        <filename>M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\Contents\AVCLIP\
009RA3LL.MXF</filename>
            <length>483</length>
        </file>
        <channels location="256">1</channels>
        <file index="0">
        <filename>M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\Contents\AVCLIP\
009RA3LL.MXF</filename>
            <length>483</length>
        </file>
        <channels location="512">1</channels>
        <file index="0">
        <filename>M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\Contents\AVCLIP\
009RA3LL.MXF</filename>
            <length>483</length>
        </file>
        <channels location="1024">1</channels>
        <file index="0">
        <filename>M:\t3media\Streams\MXF\
```

Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="2048">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="4096">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="8192">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="16384">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="32768">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\	

Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="65536">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="131072">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="262144">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="524288">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels	

location="1048576">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="2097152">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="4194304">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="8388608">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="16777216">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename>	

```
            <length>483</length>
        </file>
        <channels
location="33554432">1</channels>
        <file index="0">
        <filename>M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\Contents\AVCLIP\
009RA3LL.MXF</filename>
            <length>483</length>
        </file>
        <channels
location="67108864">1</channels>
        <file index="0">
        <filename>M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\Contents\AVCLIP\
009RA3LL.MXF</filename>
            <length>483</length>
        </file>
        <channels
location="134217728">1</channels>
        <file index="0">
        <filename>M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\Contents\AVCLIP\
009RA3LL.MXF</filename>
            <length>483</length>
        </file>
        <channels
location="268435456">1</channels>
        <file index="0">
        <filename>M:\t3media\Streams\MXF\
Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\Contents\AVCLIP\
009RA3LL.MXF</filename>
            <length>483</length>
        </file>
        <channels
location="536870912">1</channels>
        <file index="0">
        <filename>M:\t3media\Streams\MXF\
```

Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="1073741824">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="-2147483648">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="1">1</channels> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="2">1</channels> </info> <info index="52"> <file index="0"> <filename>M:\t3media\Streams\MXF\ Panasonic\Ultra-LongGop25\ PX270MP_LasVegas\Contents\AVCLIP\ 009RA3LL.MXF</filename> <length>483</length> </file> <channels location="4">1</channels>	

</info> </clipfiles> </copy>	

3.2.4 set/get – metdata (file information and metadata in XMP format)

To save the metadata for a file to a sidebar XMP filename

http://127.0.0.1:1080/netx?request=set&client=any&command=copy&profile=metadata&source=M:\t3media\Streams\MXF\Sony%20Raw\Venice%206k\C001C014_18031075_F55_X-OCN_ST_6k\C001C014_18031075.mxf

<request netxbaseip="192.168.50.100" type="command=copy"> <metadata success="1">0</metadata> <result>0</result> <resultstring>success</resultstring> </request>	***This command does not support JSON returns.***

You will probably receive a few of these ‘waiting’ returns until the media is ready. At that point, you will get back the actual cardinfo:

http://127.0.0.1:1080/netx?request=get&client=any&command=copy&profile=metadata&source=M:\t3media\Streams\MXF\Sony%20Raw\Venice%206k\C001C014_18031075_F55_X-OCN_ST_6k\C001C014_18031075.mxf

<copy> <address>127.0.0.1</address> <netxbaseip>192.168.50.100</netxbaseip> <result>0</result> <resultstring>no data from previous metadata call found yet</resultstring> </copy>	***This command does not support JSON returns.***

To return the metadata as an XML XMP via the rest API

http://127.0.0.1:1080/netx?request=get&client=any&command=copy&profile=metadata&source=M:\t3media\Streams\MXF\Sony%20Raw\Venice%206k\C001C014_18031075_F55_X-OCN_ST_6k\C001C014_18031075.mxf

See below:	***This command does not support JSON returns.***

```
<x:xmpmeta xmlns:x="adobe:ns:meta/" x:xmptk="XMP Core 5.1.2">
  <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <rdf:Description rdf:about=""
        xmlns:aux="http://ns.adobe.com/exif/1.0/aux/">
      <aux:SerialNumber>B4TCA0016</aux:SerialNumber>
    </rdf:Description>
    <rdf:Description rdf:about=""
        xmlns:tiff="http://ns.adobe.com/tiff/1.0/">
      <tiff:Model>AJ-PX270</tiff:Model>
    </rdf:Description>
    <rdf:Description rdf:about=""
        xmlns:dc="http://purl.org/dc/elements/1.1/">
      <dc:source>SHOOTING</dc:source>
    </rdf:Description>
    <rdf:Description rdf:about=""
        xmlns:xmpMM="http://ns.adobe.com/xap/1.0/mm/">

<xmpMM:DocumentID>060A2B340101010501010D431300000095EB7CB7526705CF008045822D3
24006</xmpMM:DocumentID>
      <xmpMM:VersionID>7.1.0.6</xmpMM:VersionID>
    </rdf:Description>
    <rdf:Description rdf:about=""
        xmlns:xmpDM="http://ns.adobe.com/xmp/1.0/DynamicMedia/"
        xmlns:stDim="http://ns.adobe.com/xap/1.0/sType/Dimensions#">
      <xmpDM:audioSampleRate>48000</xmpDM:audioSampleRate>
      <xmpDM:duration rdf:parseType="Resource">
        <xmpDM:value>495</xmpDM:value>
        <xmpDM:scale>1001/30000</xmpDM:scale>
      </xmpDM:duration>
      <xmpDM:videoFrameSize rdf:parseType="Resource">
        <stDim:unit>pixel</stDim:unit>
        <stDim:w>1920</stDim:w>
        <stDim:h>1080</stDim:h>
      </xmpDM:videoFrameSize>
      <xmpDM:startTimecode rdf:parseType="Resource">
        <xmpDM:timeFormat>2997DropTimecode</xmpDM:timeFormat>
        <xmpDM:timeValue>00:01:49;17</xmpDM:timeValue>
      </xmpDM:startTimecode>
      <xmpDM:altTimecode rdf:parseType="Resource">
        <xmpDM:timeFormat>2997DropTimecode</xmpDM:timeFormat>
        <xmpDM:timeValue>00:01:49;17</xmpDM:timeValue>
      </xmpDM:altTimecode>
```

```
      <xmpDM:audioChannelType>Other</xmpDM:audioChannelType>
      <xmpDM:audioSampleType>24Int</xmpDM:audioSampleType>
    </rdf:Description>
    <rdf:Description rdf:about=""
        xmlns:dt="http://www.drastictech.com/metadata/elements">
      <dt:tracks>
        <rdf:Bag>
          <rdf:li rdf:parseType="Resource">
            <rdf:value/>
            <dt:Track_Type>General</dt:Track_Type>
            <dt:Complete_name>M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\CONTENTS\AVCLIP\015KF5QO.MXF</dt:Complete_name>
            <dt:Format>MXF</dt:Format>
            <dt:Format_version>1.3</dt:Format_version>
            <dt:Format_profile>OP-1b</dt:Format_profile>
            <dt:Format_settings>Closed / Complete</dt:Format_settings>
            <dt:File_size>63.5 MiB</dt:File_size>
            <dt:Duration>16 s 517 ms</dt:Duration>
            <dt:Overall_bit_rate_mode>Variable</dt:Overall_bit_rate_mode>
            <dt:Overall_bit_rate>32.3 Mb/s</dt:Overall_bit_rate>
            <dt:Package_name>015KF5QO</dt:Package_name>
            <dt:Encoded_date>2014-04-05 17:48:53.000</dt:Encoded_date>
            <dt:Writing_application>Panasonic P2 3.0</dt:Writing_application>
            <dt:FileName>M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\CONTENTS\AVCLIP\015KF5QO.MXF</dt:FileName>
            <dt:NativeLocator>M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\CONTENTS\AVCLIP\015KF5QO.MXF</dt:NativeLocator>
            <dt:UniversalLocator>M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\CONTENTS\AVCLIP\015KF5QO.MXF</dt:UniversalLocator>
            <dt:Title>015KF5QO</dt:Title>
            <dt:Source>SHOOTING</dt:Source>
            <dt:FullName>015KF5QO</dt:FullName>
            <dt:Model>AJ-PX270</dt:Model>
            <dt:EditData3>AVC-LongG422</dt:EditData3>
            <dt:VersionString>7.1.0.6</dt:VersionString>
            <dt:Manufacturer>Panasonic</dt:Manufacturer>
            <dt:DeviceSerialNum>B4TCA0016</dt:DeviceSerialNum>
          </rdf:li>
          <rdf:li rdf:parseType="Resource">
            <rdf:value/>
            <dt:Track_Type>Video</dt:Track_Type>
            <dt:ID>2</dt:ID>
```

```
            <dt:Format>AVC</dt:Format>
            <dt:Format_Info>Advanced Video Codec</dt:Format_Info>
            <dt:Format_profile>High 4:2:2@L4</dt:Format_profile>
            <dt:Format_settings__CABAC>Yes</dt:Format_settings__CABAC>
            <dt:Format_settings__ReFrames>3 frames</dt:Format_settings__ReFrames>
            <dt:Format_settings__GOP>M=3, N=30</dt:Format_settings__GOP>
            <dt:Format_settings__wrapping_mode>Frame</dt:Format_settings__wrapping_mode>
            <dt:Codec_ID>0D01030102106001-0401020201316001</dt:Codec_ID>
            <dt:Duration>16 s 517 ms</dt:Duration>
            <dt:Bit_rate_mode>Variable</dt:Bit_rate_mode>
            <dt:Bit_rate>25.0 Mb/s</dt:Bit_rate>
            <dt:Maximum_bit_rate>37.5 Mb/s</dt:Maximum_bit_rate>
            <dt:Width>1920</dt:Width>
            <dt:Height>1080</dt:Height>
            <dt:Display_aspect_ratio>16:9</dt:Display_aspect_ratio>
            <dt:Frame_rate>29.970 (30000/1001) FPS</dt:Frame_rate>
            <dt:Standard>Component</dt:Standard>
            <dt:Color_space>YUV</dt:Color_space>
            <dt:Chroma_subsampling>4:2:2</dt:Chroma_subsampling>
            <dt:Bit_depth>10 bits</dt:Bit_depth>
            <dt:Scan_type>Interlaced</dt:Scan_type>
            <dt:Scan_type__store_method>Separated fields</dt:Scan_type__store_method>
            <dt:Scan_order>Top Field First</dt:Scan_order>
            <dt:Bits__Pixel_Frame_>0.402</dt:Bits__Pixel_Frame_>
            <dt:Stream_size>49.2 MiB (77%)</dt:Stream_size>
            <dt:Color_range>Limited</dt:Color_range>
            <dt:Color_primaries>BT.709</dt:Color_primaries>
            <dt:Transfer_characteristics>BT.709</dt:Transfer_characteristics>
            <dt:Matrix_coefficients>BT.709</dt:Matrix_coefficients>
            <dt:BitRate_Maximum_Original>37499904</dt:BitRate_Maximum_Original>
            <dt:Size>40</dt:Size>
            <dt:Planes>1</dt:Planes>
            <dt:BitCount>30</dt:BitCount>
            <dt:Compression>1635142740</dt:Compression>
            <dt:Size_Image>230400</dt:Size_Image>
            <dt:TimeCode>3285</dt:TimeCode>
            <dt:VITCTimeCode>3285</dt:VITCTimeCode>
            <dt:VersionNumber>70100006</dt:VersionNumber>
            <dt:TimeCodeType>4</dt:TimeCodeType>
            <dt:LTCTimeCodeType>4</dt:LTCTimeCodeType>
            <dt:VITCTimeCodeType>4</dt:VITCTimeCodeType>
          </rdf:li>
```

```
        <rdf:li rdf:parseType="Resource">
          <rdf:value/>
          <dt:Track_Type>Audio</dt:Track_Type>
          <dt:ID>3</dt:ID>
          <dt:Format>PCM</dt:Format>
          <dt:Format_settings__Endianness>Little</dt:Format_settings__Endianness>
          <dt:Codec_ID>0D01030102060B00</dt:Codec_ID>
          <dt:Duration>16 s 517 ms</dt:Duration>
          <dt:Bit_rate_mode>Constant</dt:Bit_rate_mode>
          <dt:Bit_rate>1 152 kb/s</dt:Bit_rate>
          <dt:Channel_s_>1 channel</dt:Channel_s_>
          <dt:Sampling_rate>48.0 kHz</dt:Sampling_rate>
          <dt:Bit_depth>24 bits</dt:Bit_depth>
          <dt:Stream_size>2.27 MiB (4%)</dt:Stream_size>
          <dt:Locked>Yes</dt:Locked>
          <dt:Format_Tag>1</dt:Format_Tag>
          <dt:Channels>4</dt:Channels>
          <dt:Samples_Per_Sec>48000</dt:Samples_Per_Sec>
          <dt:Avg_Bytes_Per_Sec>576000</dt:Avg_Bytes_Per_Sec>
          <dt:Block_Align>16</dt:Block_Align>
          <dt:Bits_Per_Sample>24</dt:Bits_Per_Sample>
          <dt:Reserved>1935963489</dt:Reserved>
          <dt:ccType>1935963489</dt:ccType>
          <dt:ccHandler>1</dt:ccHandler>
          <dt:Scale>12</dt:Scale>
          <dt:Rate>576000</dt:Rate>
          <dt:Length>792792</dt:Length>
          <dt:Suggested_Buffer_Size>288000</dt:Suggested_Buffer_Size>
          <dt:Sample_Size>12</dt:Sample_Size>
          <dt:File_Type>172</dt:File_Type>
        </rdf:li>
        <rdf:li rdf:parseType="Resource">
          <rdf:value/>
          <dt:Track_Type>Audio</dt:Track_Type>
          <dt:ID>4</dt:ID>
          <dt:Format>PCM</dt:Format>
          <dt:Codec_ID>0D01030102060B00</dt:Codec_ID>
          <dt:Duration>16 s 517 ms</dt:Duration>
          <dt:Bit_rate>1 152 kb/s</dt:Bit_rate>
          <dt:Channel_s_>1 channel</dt:Channel_s_>
          <dt:Sampling_rate>48.0 kHz</dt:Sampling_rate>
          <dt:Bit_depth>24 bits</dt:Bit_depth>
```

```
      <dt:Stream_size>2.27 MiB (4%)</dt:Stream_size>
      <dt:Locked>Yes</dt:Locked>
    </rdf:li>
    <rdf:li rdf:parseType="Resource">
      <rdf:value/>
      <dt:Track_Type>Audio</dt:Track_Type>
      <dt:ID>5</dt:ID>
      <dt:Format>PCM</dt:Format>
      <dt:Codec_ID>0D01030102060B00</dt:Codec_ID>
      <dt:Duration>16 s 517 ms</dt:Duration>
      <dt:Bit_rate>1 152 kb/s</dt:Bit_rate>
      <dt:Channel_s_>1 channel</dt:Channel_s_>
      <dt:Sampling_rate>48.0 kHz</dt:Sampling_rate>
      <dt:Bit_depth>24 bits</dt:Bit_depth>
      <dt:Stream_size>2.27 MiB (4%)</dt:Stream_size>
      <dt:Locked>Yes</dt:Locked>
    </rdf:li>
    <rdf:li rdf:parseType="Resource">
      <rdf:value/>
      <dt:Track_Type>Audio</dt:Track_Type>
      <dt:ID>7</dt:ID>
      <dt:Format>PCM</dt:Format>
      <dt:Codec_ID>0D01030102060B00</dt:Codec_ID>
      <dt:Duration>16 s 517 ms</dt:Duration>
      <dt:Bit_rate>1 152 kb/s</dt:Bit_rate>
      <dt:Channel_s_>1 channel</dt:Channel_s_>
      <dt:Sampling_rate>48.0 kHz</dt:Sampling_rate>
      <dt:Bit_depth>24 bits</dt:Bit_depth>
      <dt:Stream_size>2.27 MiB (4%)</dt:Stream_size>
      <dt:Locked>Yes</dt:Locked>
    </rdf:li>
    <rdf:li rdf:parseType="Resource">
      <rdf:value/>
      <dt:Track_Type>Other</dt:Track_Type>
      <dt:ID>1-Material</dt:ID>
      <dt:Type>Time code</dt:Type>
      <dt:Format>MXF TC</dt:Format>
      <dt:Time_code_of_first_frame>00:01:49;17</dt:Time_code_of_first_frame>
      <dt:Time_code_settings>Material Package</dt:Time_code_settings>
      <dt:Time_code__striped>Yes</dt:Time_code__striped>
    </rdf:li>
    <rdf:li rdf:parseType="Resource">
```

```
            <rdf:value/>
            <dt:Track_Type>Other</dt:Track_Type>
            <dt:ID>1-Source</dt:ID>
            <dt:Type>Time code</dt:Type>
            <dt:Format>MXF TC</dt:Format>
            <dt:Time_code_of_first_frame>00:01:49;17</dt:Time_code_of_first_frame>
            <dt:Time_code_settings>Source Package</dt:Time_code_settings>
            <dt:Time_code__striped>Yes</dt:Time_code__striped>
          </rdf:li>
        </rdf:Bag>
      </dt:tracks>
      <dt:CreationData>2014-04-05T17:49:10+00:00</dt:CreationData>
      <dt:EditData3>AVC-LongG422</dt:EditData3>
      <dt:FileName>M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\PX270MP_LasVegas\
CONTENTS\AVCLIP\015KF5QO.MXF</dt:FileName>
      <dt:DeviceModelNum>AJ-PX270</dt:DeviceModelNum>
      <dt:DeviceSerialNum>B4TCA0016</dt:DeviceSerialNum>
      <dt:FullName>015KF5QO</dt:FullName>
      <dt:Manufacturer>Panasonic</dt:Manufacturer>
      <dt:Model>AJ-PX270</dt:Model>
      <dt:NativeLocator>M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\CONTENTS\AVCLIP\015KF5QO.MXF</dt:NativeLocator>
      <dt:Source>SHOOTING</dt:Source>
      <dt:Title>015KF5QO</dt:Title>
      <dt:UniversalLocator>M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\
PX270MP_LasVegas\CONTENTS\AVCLIP\015KF5QO.MXF</dt:UniversalLocator>
      <dt:VersionString>7.1.0.6</dt:VersionString>
      <dt:A-Frame>0</dt:A-Frame>
      <dt:PosterFrame>0</dt:PosterFrame>
      <dt:TimeCode>3285</dt:TimeCode>
      <dt:UserBits>335807767</dt:UserBits>
      <dt:VITCTimeCode>3285</dt:VITCTimeCode>
      <dt:VITCUserBits>335807767</dt:VITCUserBits>
      <dt:VersionNumber>70100006</dt:VersionNumber>
      <dt:TimeCodeType>4</dt:TimeCodeType>
      <dt:LTCTimeCodeType>4</dt:LTCTimeCodeType>
      <dt:VITCTimeCodeType>4</dt:VITCTimeCodeType>
    </rdf:Description>
    <rdf:Description rdf:about=""
        xmlns:dtdev="http://www.drastictech.com/metadata/deviceelements">
      <dtdev:DTDeviceMetadataType>PANASONICP2</dtdev:DTDeviceMetadataType>
      <dtdev:ClipContent-
```

```
GlobalClipID>060A2B340101010501010D431300000095EB7CB7526705CF008045822D324006</
dtdev:ClipContent-GlobalClipID>
        <dtdev:ClipContent-Duration>495</dtdev:ClipContent-Duration>
        <dtdev:ClipContent-EditUnit>1001/30000</dtdev:ClipContent-EditUnit>
        <dtdev:Video-Codec>AVC-LongG422</dtdev:Video-Codec>
        <dtdev:Video-ActiveLine>1080</dtdev:Video-ActiveLine>
        <dtdev:Video-FrameRate>59.94i</dtdev:Video-FrameRate>
        <dtdev:Video-TimecodeType>Drop</dtdev:Video-TimecodeType>
        <dtdev:Video-StartTimecode>00:01:49:17</dtdev:Video-StartTimecode>
        <dtdev:Video-EndTimecode>00:02:06:03</dtdev:Video-EndTimecode>
        <dtdev:Video-StartBinaryGroup>14040517</dtdev:Video-StartBinaryGroup>
        <dtdev:Audio-Channel>4</dtdev:Audio-Channel>
        <dtdev:Audio-SamplingRate>48000</dtdev:Audio-SamplingRate>
        <dtdev:Audio-BitsPerSample>24</dtdev:Audio-BitsPerSample>
        <dtdev:ClipMetadata-UserClipName>015KF5QO</dtdev:ClipMetadata-UserClipName>
        <dtdev:ClipMetadata-DataSource>SHOOTING</dtdev:ClipMetadata-DataSource>
        <dtdev:Access-CreationDate>2014-04-05T17:48:53+00:00</dtdev:Access-CreationDate>
        <dtdev:Access-LastUpdateDate>2014-04-05T17:49:10+00:00</dtdev:Access-
LastUpdateDate>
        <dtdev:Device-Manufacturer>Panasonic</dtdev:Device-Manufacturer>
        <dtdev:Device-SerialNo.>B4TCA0016</dtdev:Device-SerialNo.>
        <dtdev:Device-ModelName>AJ-PX270</dtdev:Device-ModelName>
        <dtdev:Shoot-StartDate>2014-04-05T17:48:53+00:00</dtdev:Shoot-StartDate>
        <dtdev:Shoot-EndDate>2014-04-05T17:49:10+00:00</dtdev:Shoot-EndDate>
        <dtdev:Thumbnail-FrameOffset>0</dtdev:Thumbnail-FrameOffset>
        <dtdev:Thumbnail-ThumbnailFormat>BMP</dtdev:Thumbnail-ThumbnailFormat>
        <dtdev:Thumbnail-Width>80</dtdev:Thumbnail-Width>
        <dtdev:Thumbnail-Height>45</dtdev:Thumbnail-Height>
        <dtdev:Proxy-ProxyFormat>MOV</dtdev:Proxy-ProxyFormat>
        <dtdev:ProxyVideo-ProxyVideoCodec>AVC</dtdev:ProxyVideo-ProxyVideoCodec>
        <dtdev:ProxyVideo-ProxyVideoBitRate>6000000</dtdev:ProxyVideo-ProxyVideoBitRate>
        <dtdev:ProxyVideo-ProxyFrameRate>29.97</dtdev:ProxyVideo-ProxyFrameRate>
        <dtdev:ProxyVideo-ProxyResolution>1920x1080</dtdev:ProxyVideo-ProxyResolution>
        <dtdev:ProxyVideo-AspectRatio>16:9</dtdev:ProxyVideo-AspectRatio>
        <dtdev:ProxyVideo-OnScreenTimecode>OFF</dtdev:ProxyVideo-OnScreenTimecode>
        <dtdev:ProxyAudio-ProxyAudioCodec>AAC</dtdev:ProxyAudio-ProxyAudioCodec>
        <dtdev:ProxyAudio-ProxyAudioBitRate>64000</dtdev:ProxyAudio-ProxyAudioBitRate>
        <dtdev:ProxyAudio-ProxySamplingRate>48000</dtdev:ProxyAudio-ProxySamplingRate>
        <dtdev:ProxyAudio-OriginalChannel>1</dtdev:ProxyAudio-OriginalChannel>
    </rdf:Description>
  </rdf:RDF>
</x:xmpmeta>
```

This command also supports writing the metadata to a disk locations with.

```
http://127.0.0.1:1080/netx?
request=set&client=any&command=copy&profile=metadata&source=M:\t3media\Streams\MXF\
Sony%20Raw\Venice%206k\C001C014_18031075_F55_X-OCN_ST_6k\
C001C014_18031075.mxf&target=m:\t3media\metadata\C001C014_18031075.mxf.xmp
```

```
<copy>
  <address>192.168.100.229</address>
  <netxbaseip>192.168.50.100</netxbaseip>
  <result>0</result>
  <resultstring>no data from previous cardinfo call found yet</resultstring>
</copy>
```

```
{
  "address": "192.168.100.229",
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "no data from previous metadata call found yet"
}
```

NOTE: The XMP output is compatible with the XMP standard. There are two extensions included to allow for more information than is normally available in an XMP file. The first is the ‘dt: space that includes extended track and metadata information. It has each of the files’ ‘tracks’ set in an array of ‘li’ elements with a ‘bag’ within the document. The structure of the tracks is

```
<rdf:Description rdf:about=""
        xmlns:dt="http://www.drastictech.com/metadata/elements">

      <dt:tracks>
        <rdf:Bag>
          <rdf:li rdf:parseType="Resource">
… track info
          </rdf:li>
          <rdf:li rdf:parseType="Resource">
… track info
          </rdf:li>
          <rdf:li rdf:parseType="Resource">
          </rdf:li>
        </rdf:Bag>
      </dt:tracks>
```

After that there will be a number of general metadata parameters.

The second section is the ‘dtdev:’. This is a device specific area that passes back advanced parameters from the file, if available. This is commonly used for the shooting parameters of cameras from RED, Sony, Panasonic, Canon, Blackmagic and others. It will contain any available parameters set in the camera, usually including items like colorimetry, camera setting, GPS coordinates, model/serial number and shoot/take/reel information.

3.2.4.1 Parameters:

command=copy – must be copy to use the copy/transcode/PFR
source=<sourcefile> – source name may be direct path, unc path or other system accessible path
target=<targetfile> – target name may be direct path, unc path or other system accessible path
ack=<ackfile> – the XML ACK file that will be created on completion with status and metadata
tcin=01:00:00:00 – start as a time code
tcout=01:01:00:00 – end as a time code reference (exclusive)
absin=00:00:10:00 – first frame number in absolute frames from 0. Also supports frame number
absout=450 – last frame number in absolute frames from 0 (exclusive). Also supports time code
tcoffset=01:00:00:00 – amount to offset the time code of the output clip. Does not affect the tcin/tcout.
copy - make a copy of the file section we need before translating, instead of reading directly
dest=<destfolder> - folder or folder and file name for the temp file when using copy
profile=*** - Profile to use.

Current profiles include:

copy - copy the whole file
wrap - re wrap file or part of a file
index - create an RTIndex for a file
getCopyInOut - get the extents required for a pfr, or use them with a temp file
mp3-128kbps - Audio MP3 file
mov-YCbCr8Bit - QuickTime MOV 8 bit uncompressed YCbCr file
mov-dvcprohd - QuickTime MOV DVCPro HD (1080/720)
mp4-h264 - MPEG-4 h264 AAC Audio
mxf-xdcam-720p - True XDCam MXF 8 channel audio
mxf-dvcprohd-720p - MXF DVCPro HD 720p
mxf-xdcam-1080i - True XDCam MXF 1080i 8 channel audio
mxf-dvcprohd-1080i - MXF DVCPro HD 1080i 29/25 fps
mxf-OP1a-MPEG - OpenMXF XDCam MPEG-2 16 channel audio
mxf-OP1a-h264 - MXF h.264
mxf-OP1a-HDF - MXF MPEG-2 HDF Standard
mxf-as-11-sd-pal-dpp - MXF AS-11 SD PAL DPP
mxf-as-11-sd-ntsc-dpp - MXF AS-11 SD NTSC DPP
mxf-as-11-hd-dpp - MXF DPP AS-11 AVCi HD
mov-proreshq - QuickTime MOV ProRes HQ
mov-proreslt - QuickTime MOV ProRes LT
mov-prores422 - QuickTime MOV ProRes 422
mov-prores444 - QuickTime MOV ProRes 444(4)
scaledown2000k - MP4 264 960x540, 2mbs, AAC

scaledown500k - MP4 264 480x272, 0.5mbs, AAC
hd1080-5000kbs - MP4 HD 1080 with a target bitrate of 5 mbs
hd720-2500kbs - MP4 HD 720p with a target bitrate of 2.5 mbs
hd360-1250kbs - MP4 HD 360p with a target bitrate of 1.25 mbs
h264-7500kbs - MP4 Any resolution with a target bitrate of 7.5 mbs
Proxy-h264-5000kbs - MP4 high quality proxy for web
LBR-h264-10000kbs - Low bit rate, high quality local MP4
mxf-OP1a-JPEG2K - Samma style JPEG2000 YCbCr
mxf-AS-02-h264-10 - 10 bit 50 Mbs h.264 in AS-02 MXF
DASH-MP4-Multibitrate - Multi bitrate MP4s with DASH files
HLS-TS-Multibitrate - Multi bitrate TS streams with M3U8 files
MP4-MultiOutput - Multi MP4 with optional burn in files
TS-TR-01-JPEG-2000 - TR-01 JPEG-2000 transport stream
TS-MPEG2 - MPEG-2 4:2:0/passthrough transport stream
TS-h264 - h.264 4:2:0/passthrough transport stream
OP1a_HBR_50 - OP1a MXF h264 4:2:2 10 bit
mp4-XAVC-S_4_2_0 - MP4 Sony XAVC-S 4:2:0
mp4-XAVC-S_4_2_2 - MP4 Sony XAVC-S 4:2:2
aces - ACES image files
dnxhd-mxf-720p - DNxHD 720p 50, 59, 60
dnxhd-mxf-1080p - DNxHD 1080p 25, 29
dnxhd-mxf-1080i - DNxHD 1080i 25, 29
dnxhr-mxf-10-hq - DNxHR High Quality 10 bit
dnxhr-mxf-8-hq - DNxHR High Quality 8 bit
dnxhr-mxf-sq - DNxHR Standard Quality
dnxhr-mxf-lq - DNxHR Low Quality
type=* - the exact file type to write, otherwise auto
mxf-op1a - standard OP1a
mxf-sonyhd - Sony XDCam compatible
mxf-as02 - AS-02 spec MXF
mxf-open - Edit while ingest MXF
mp4-fmp4 - Fragmented MP4 (normal MP4 if not set)
mov - QuickTime MOV
metadata[=optional file name] - Save metadata from a file to an XML/XMP file
Width=480 - Sets the width of the picon or video output, when not a fixed size codec (like DVHD)
Height=272 - Sets the height of the video output, when not a fixed size codec (like DVHD). Ha no effect on picons.
cc=<ccfile> - optional alternate closed caption file (SCC or MCC preferred)
afile=<audiofile> - alternate audio file source (aiff or wav)
video=<videofile> - alternate video file source
stereo - sum all the audio channels into a single stereo pair
aroute=78123456 - route any audio channel from any audio channel

uuid=<uuid> - apply a specific UUID to the output filename
kilobitrate=5000 - override the default compression bit rate with this rate (mbs * 1024)
h26xprofile=41 - use this profile for h264/h265 (see below)
h26xlevel=77 - use this level for h264/h265 (see below)
gopsize=15 - number of frames in a GOP
encodemode=0 - fast (1) or quality (0) encoding
tempfolder=<tempdir> - temporary folder to hold the tape PFR binary chunk
alignment=4096 - required alignment of the PFR tape or disk source
flags – see below
fg - force the NetXCopy GUI to be displayed
fc - force the command line to be used
piSrc = Source file to be used to create a picon JPEG image
piDst = Destination file for the JPEG picon image
piSize = Percentage size for the JPEG, integer 1..100% (e.g. 1920x1080 @ 17 would be 320x180)
piFrame = Either the absolute frame value (integer) or a time code string of the frame to use
piSkip = Number of frames to skip between picons. If set, a series of picons will be produced at 'skip' intervals
width = set the width of the output picon. The height will be calculated from this

h26x Profiles:

10, 11, 12, 13, 20, 21, 22, 30, 31, 32, 40, 41, 42, 50, 51, 52

h26x Levels:

66 – Baseline
77 – Main
88 – Extended
100 – High
110 – High 10
122 – High 4:2:2
144 – High 4:4:4
166 – Advanced 4:4:4 Intra
188 – Advanced 4:4:4

flags – optional flags for conversions

allSubAudioTracks – search for all the audio tracks in a source file
aiProfileID – the eMAM AI Profile ID for transcript insertion
ccExtractChannel – which CC# (608) or service# (708) to extract
ccOutputDirectory – directory to write the output CC to
convert708to608 – down convert the 708 to 608, otherwise extract the compatibility bytes
eMamAssetID=# - the asset ID to patch in eMAM
eMamCategory= - the category to assign the item to in eMAM
eMamProxyID=# - the proxy ID to patch in eMAM

extractCCLine21 – force decoding the visual encoded line 21 video (SD)
fixOut – if the out is not clean, just continue to the end
forceCCVidStream=<videofile> - use this video file for the CC insert output
forceOutputFramerate – override the framerate of the output file
fpsCompensation – compensate for changing fps. 23.98 to 29.97 would be 2329
frameAccurate – re encode head and/or tail of interframe compressed files, otherwise cut at nearest whole frame-offset-into-file
interplay – pass the interplay project to assign this conversion into
interpretFileEncoding= - force a particular file encoding
markFirstTime – mark the first timecode of the caption file
multi# - use multi channel audio blocks
notifyeMam – send a notification to an eMAM server's using eMamAssetID etc.
offsetTC=00:01:00:00 – offset the original timecode by this much
overrideTC=01:00:00:00 – override the native timecode starting with this one
proxyAfterRecord – not available yet
proxyDisable – not available yet
proxyEnable – not available yet
proxyRecord – not available yet
shuffleAudio – reshuffle the audio channels based on the internal markings to make the left/right sum as the first pair, followed by everything else
testMode – test the conversion
timeConversion – output how long the conversion took
zeroBasedOutput – output the caption file starting at 00:00:00:00

Set can also be used to cancel and/or clear processes and information.

To set the max number of copy operations for the client:

http://127.0.0.1:1080/netx?request=set&client=0&command=copy&copylimit=5

<request netxbaseip="192.168.50.100"
type="command=copy">
<result>0</result>
<resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "type": "command=copy"
}
```


To clear what is in the current working information, call:

http://127.0.0.1:1080/netx?request=set&client=0&command=copy&clear=true

<request netxbaseip="192.168.50.100"
type="command=copy">


```
{
  "netxbaseip": "192.168.50.100",
```

<result>0</result> <resultstring>success</resultstring> </request>	"result": "0", "resultstring": "success", "type": "command=copy" }

To abort the current process, call:

http://127.0.0.1:1080/netx?request=set&client=0&command=copy&abort=true

<request netxbaseip="192.168.50.100"
type="command=copy">
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "success",
	"type": "command=copy"
	}

To clean out the copy/convert history for all items, call:

http://127.0.0.1:1080/netx?request=set&client=0&group=-1&command=copy&purge=true

<request netxbaseip="192.168.50.100"
type="command=copy">
<result>-2</result>
<resultstring>failure</resultstring>
</request>
	{
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "success",
	"type": "command=copy"
	}

To clean out the copy/convert history for a single item, call as you would for the purge trigger and specify the source, target, and profile.

The triggers for clear, purge and abort can all be used simultaneously for varied effects:

abort + clear will abort the current process and clear out any pending copy/conversions.

3.2.5 copylimit

Set the maximum number of simultaneous copies allowed on one server.

http://127.0.0.1:1080/netx?request=set&client=0&group=1&command=copy&copylimit=5

<request netxbaseip="192.168.50.100" type="command=copy">	{ "netxbaseip": "192.168.50.100",

<result>0</result> <resultstring>success</resultstring> </request>	"result": "0", "resultstring": "success", "type": "command=copy" }

3.2.6 get - status/completion

Is used to obtain information about current or past copy requests. There are two queues of information; the active copies and the completed copies. To get the completed copies, add 'complete=1' to the request. If the source and target parameters are not set, then information about the current copy request is returned. If they are specified, then the system will try to locate information about the specific copy/conversion from memory. (NOTE: if a purge request was made then the specified item may no longer exist in memory)

```
http://127.0.0.1:1080/netx?request=set&client=0&group=-1&command=copy&profile=mxf-OP1a-MPEG&source=d:\record\first_file.mxf&target=d:\record\second_file.mxf
```

A simple get to obtain information about the current process would be as follows (note that not specifying a group is the same as specifying the -1 all group):

```
http://127.0.0.1:1080/netx?request=get&client=0&command=copy
```

<copy>
<address>192.168.100.189</address>
 	
<netxbaseip>192.168.100.189</netxbaseip>
<copyid index="0" id="34952">
<source>
M:\t3media\Streams\MXF\Panasonic\Ultra-LongGop25\PX270MP_LasVegas\
</source>
<target>2</target>
<targetrtin>2.rtin</targetrtin>
<targetack>2.ack</targetack>
<profile>cardinfo</profile>
<tcoffset>0</tcoffset>
<type>copy</type>
<progress>0</progress>
<startms>91823761</startms>
<currentms>0</currentms>
<completed>0</completed>
	{
	"address": "192.168.100.189",
	"copyid0": {
	"completed": "0",
	"currentms": "0",
	"id": "34952",
	"index": "0",
	"profile": "cardinfo",
	"progress": "0",
	"result": "0",
	"resultstring": "Copy Pending",
	"source": "M:\\t3media\\Streams\\MXF\\Panasonic\\Ultra-LongGop25\\PX270MP_LasVegas\\",
	"startms": "91823761",
	"target": "2",
	"targetack": "2.ack",
	"targetrtin": "2.rtin",
	"tcoffset": "0",

<result>0</result>
<resultstring>Copy Pending</resultstring>
</copyid>
<copyid index="1" id="34952">
<source>M:\t3media\Streams\MXF\Sony Raw\Venice 6k\C001C014_18031075_F55_X-OCN_ST_6k\C001C014_18031075.mxf</source>
<target>m:\t3media\metadata\C001C014_18031075.mxf.xmp</target>
<targetrtin>m:\t3media\metadata\C001C014_18031075.mxf.rtin</targetrtin>
<targetack>m:\t3media\metadata\C001C014_18031075.mxf.ack</targetack>
<profile>metadata</profile>
<tcoffset>0</tcoffset>
<type>copy</type>
<progress>0</progress>
<startms>92124504</startms>
<currentms>0</currentms>
<completed>0</completed>
<result>0</result>
<resultstring>Copy Pending</resultstring>
</copyid>
<result>0</result>
<resultstring>returning requested copy</resultstring>
</copy>


```
  "type": "copy"
 },
 "copyid1": {
  "completed": "0",
  "currentms": "0",
  "id": "34952",
  "index": "1",
  "profile": "metadata",
  "progress": "0",
  "result": "0",
  "resultstring": "Copy Pending",
  "source": "M:\\t3media\\Streams\\MXF\\Sony Raw\\Venice 6k\\C001C014_18031075_F55_X-OCN_ST_6k\\C001C014_18031075.mxf",
  "startms": "92124504",
  "target": "m:\\t3media\\metadata\\C001C014_18031075.mxf.xmp",
  "targetack": "m:\\t3media\\metadata\\C001C014_18031075.mxf.ack",
  "targetrtin": "m:\\t3media\\metadata\\C001C014_18031075.mxf.rtin",
  "tcoffset": "0",
  "type": "copy"
 },
 "netxbaseip": "192.168.100.189",
 "result": "0",
 "resultstring": "returning requested copy"
}
```


http://127.0.0.1:1080/netx?request=get&client=0&group=-1&command=copy


```
<copy>
<address>192.168.100.189</address>
<netxbaseip>192.168.100.189</netxbaseip>
<copyid index="0" id="34952">
<source>M:\t3media\Streams\MXF\Panasonic\Ultra-
```

```
{
 "address": "192.168.100.189",
 "copyid0": {
  "completed": "0",
  "currentms": "0",
  "id": "34952",
  "index": "0",
```

```
LongGop25\PX270MP_LasVegas\
                </source>
                <target>2</target>
                <targetrtin>2.rtin</targetrtin>
                <targetack>2.ack</targetack>
                <profile>cardinfo</profile>
                <tcoffset>0</tcoffset>
                <type>copy</type>
                <progress>0</progress>
                <startms>91823761</startms>
                <currentms>0</currentms>
                <completed>0</completed>
                <result>0</result>
                <resultstring>Copy
Pending</resultstring>
        </copyid>
        <copyid index="1" id="34952">
                <source>
M:\t3media\Streams\MXF\Sony Raw\Venice 6k\
C001C014_18031075_F55_X-OCN_ST_6k\
C001C014_18031075.mxf
                </source>
                <target>m:\t3media\metadata\
C001C014_18031075.mxf.xmp</target>
                <targetrtin>m:\t3media\metadata\
C001C014_18031075.mxf.rtin</targetrtin>
                <targetack>m:\t3media\metadata\
C001C014_18031075.mxf.ack</targetack>
                <profile>metadata</profile>
                <tcoffset>0</tcoffset>
                <type>copy</type>
                <progress>0</progress>
                <startms>92124504</startms>
                <currentms>0</currentms>
                <completed>0</completed>
                <result>0</result>
                <resultstring>Copy
Pending</resultstring>
        </copyid>
        <result>0</result>
        <resultstring>returning requested
copy</resultstring>
```

```
    "profile": "cardinfo",
    "progress": "0",
    "result": "0",
    "resultstring": "Copy Pending",
    "source": "M:\\t3media\\Streams\\MXF\\
Panasonic\\Ultra-LongGop25\\
PX270MP_LasVegas\\",
    "startms": "91823761",
    "target": "2",
    "targetack": "2.ack",
    "targetrtin": "2.rtin",
    "tcoffset": "0",
    "type": "copy"
  },
  "copyid1": {
    "completed": "0",
    "currentms": "0",
    "id": "34952",
    "index": "1",
    "profile": "metadata",
    "progress": "0",
    "result": "0",
    "resultstring": "Copy Pending",
    "source": "M:\\t3media\\Streams\\MXF\\Sony
Raw\\Venice 6k\\C001C014_18031075_F55_X-
OCN_ST_6k\\C001C014_18031075.mxf",
    "startms": "92124504",
    "target": "m:\\t3media\\metadata\\
C001C014_18031075.mxf.xmp",
    "targetack": "m:\\t3media\\metadata\\
C001C014_18031075.mxf.ack",
    "targetrtin": "m:\\t3media\\metadata\\
C001C014_18031075.mxf.rtin",
    "tcoffset": "0",
    "type": "copy"
  },
  "netxbaseip": "192.168.100.189",
  "result": "0",
  "resultstring": "returning requested copy"
}
```

</copy>	

Alternately, a copy request may be found by specifying the complete target of the copy request and setting the group to 'find':

http://127.0.0.1:1080/netx?request=get&client=0&command=copy&group=find&target=d:\record\second_file

<copy>
<address>192.168.100.229</address>
<netxbaseip>192.168.50.100</netxbaseip>
<copyid id="-1" index=”0”>
<source>d:\record\first_file.mxf</source>
<target/>
<targetrtin>d:\record\first_file .rtin</targetrtin>
<targetack>d:\record\first_file .ack</targetack>
<profile>mxf-OP1a-MPEG</profile>
<tcoffset/>
<completed>0</completed>
<progress>535</progress>
<startms>100345709</startms>
<currentms>-1</currentms>
<type/>
<message>Copy Pending</message>
</copyid>
<result>0</result>
<resultstring>returning requested copy</resultstring>
</copy>

{
"address": "192.168.100.229",
"copyid0": {
"currentms": "",
"id": "-1",
“index”: “0”,
"profile": "mxf-OP1a-MPEG",
“completed”: “1”,
"progress": "10000",
"source": "d:\\record\\first_file.mxf",
"startms": "",
"target": "d:\\record\\second_file.mxf",
"targetack": "d:\\record\\second_file.ack",
"targetrtin": "d:\\record\\second_file.rtin",
"tcoffset": "0",
"type": "copy"
},
"netxbaseip": "192.168.50.100",
"result": "0",
"resultstring": "returning requested copy"
}

To get the status of a specific active copy, specify the running copy as the group value. For copies, group=-1 means all copies. Group=-2 or less specifies a specific copy. The maximum total copies on a server is dependent on how many have been sent and what the copy limit has been set to.

http://127.0.0.1:1080/netx?request=get&client=0&group=-1&command=copy

<copy>
<address>192.168.100.229</address>
<netxbaseip>192.168.50.100</netxbaseip>
<copyid id="-1" index=”1”>
<source>d:\record\first_file.mxf</source>
<target/>
<targetrtin>.rtin</targetrtin>

{
"address": "192.168.100.229",
"copyid1": {
"currentms": "168547725",
"id": "-2",
“index”: “1”,
"profile": "wrap",

<targetack>.ack</targetack> <profile>mxf-OP1a-MPEG</profile> <tcoffset/> <type/> <progress>0</progress> <completed>0</completed> <startms>100345709</startms> <currentms>-1</currentms> <message>Copy Pending</message> </copyid> <copyid id="-2" index="1"> <source>d:\record\first_file.mxf</source> <target>d:\record\second_file.mxf</target> <targetrtin>d:\record\second_file.rtin</targetrtin> <targetack>d:\record\second_file.ack</targetack> <profile>mxf-OP1a-MPEG</profile> <tcoffset/> <type/> <completed>0</completed> <progress>6783</progress> <startms>100346586</startms> <currentms>100460047</currentms> </copyid> <result>0</result> <resultstring>returning requested copy</resultstring> </copy>	"completed": "0", "progress": "4153", "source": "M:\\t3media\\Streams\\NBA\\NBA_Reference_Test_Files\\amberfin_new_0000750142.rtin", "startms": "166595260", "target": "m:\\restorepoint\\partial\\amberfin_clip.mxf", "targetack": "m:\\restorepoint\\partial\\amberfin_clip.ack", "targetrtin": "m:\\restorepoint\\partial\\amberfin_clip.rtin", "tcoffset": "0", "type": "copy" }, "netxbaseip": "192.168.50.100", "result": "0", "resultstring": "returning requested copy" }

A get with the id, source, target and profile specified will get what is known for that particular copy/conversion.

To get a list of the completed copy requests, add the complete=1 flag

http://127.0.0.1:1080/netx?request=get&client=0&command=copy&complete=1

<copy>
<address>192.168.100.229</address>
<netxbaseip>192.168.50.100</netxbaseip>
<result>0</result>
<resultstring>returning requested copy</resultstring>
	{
	"address": "192.168.100.229",
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "returning requested copy"
	}

</copy>	

By default, calls to the copy complete do not purge the complete status. This is so that multiple callers can all get the complete for a single file. To remove the complete, a 'purge' must be added.

http://127.0.0.1:1080/netx?request=get&client=0&command=copy&complete=1&purge=1

<copy>
<address>192.168.100.229</address>
<netxbaseip>192.168.50.100</netxbaseip>
<copyid id="-2" index="0">
<source>d:\record\first_file.mxf</source>
<target>d:\record\second_file.mxf</target>

<targetrtin>d:\record\second_file.rtin</targetrtin>

<targetack>d:\record\second_file.ack</targetack>
<profile>mxf-OP1a-MPEG</profile>
<tcoffset/>
<type/>
<completed>1</complete>
<progress>10000</progress>
<startms/>
<currentms/>
<message>Copy successful</message>
</copyid>
<result>0</result>
<resultstring>returning requested copy</resultstring>
</copy>

{
"address": "192.168.100.229",
"netxbaseip": "192.168.50.100",
"result": "0",
"resultstring": "returning requested copy"
}

To get a list of the active copy requests, the command would be:

http://127.0.0.1:1080/netx?request=get&client=0&command=copy

<copy>
<address>192.168.100.229</address>
<netxbaseip>192.168.50.100</netxbaseip>
<copyid id="-1" index="0">
<source>d:\record\first_file.mxf</source>
<target/>
<targetrtin>.rtin</targetrtin>
<targetack>.ack</targetack>

{
"address": "192.168.100.229",
"copyid0": {
"currentms": "168637148",
"id": "-2",
"index": "0",
"profile": "wrap",
"completed": "0",

<profile>mxf-OP1a-MPEG</profile> <tcoffset/> <type/> <completed>0</completed> <progress>0</progress> <startms>100345709</startms> <currentms>-1</currentms> <message>Copy Pending</message> </copyid> <copyid id="-2"> <source>d:\record\first_file.mxf</source> <target>d:\record\second_file.mxf</target> <targetrtin>d:\record\second_file.rtin</targetrtin> <targetack>d:\record\second_file.ack</targetack> <profile>mxf-OP1a-MPEG</profile> <tcoffset/> <type/> <progress>-1</progress> <startms/> <currentms/> <message>Copy successful</message> </copyid> <result>0</result> <resultstring>returning requested copy</resultstring> </copy>	"progress": "4346", "source": "M:\\t3media\\Streams\\NBA\\NBA_Reference_Test_Files\\amberfin_new_0000750142.rtin", "startms": "166595260", "target": "m:\\restorepoint\\partial\\amberfin_clip.mxf", "targetack": "m:\\restorepoint\\partial\\amberfin_clip.ack", "targetrtin": "m:\\restorepoint\\partial\\amberfin_clip.rtin", "tcoffset": "0", "type": "copy" }, "netxbaseip": "192.168.50.100", "result": "0", "resultstring": "returning requested copy" }

If the client is unknown, it can be found by using client=any with the target specified by the copy request.

http://127.0.0.1:1080/netx?request=get&client=any&group=find&command=copy&target=d:\record\second_file

<copy>
<address>192.168.100.229</address>
<netxbaseip>192.168.50.100</netxbaseip>
<copyid id="-1" index=”0”>
<source>d:\record\first_file.mxf</source>
<target/>
<targetrtin>d:\record\first_file.rtin</targetrtin>
<targetack>d:\record\first_file.ack</targetack>
	{
	"address": "192.168.100.229",
	"copyid0": {
	"id": "-1",
	“index”: “0”,
	"message": "copy not found",
	"profile": "",
	"source": "",

<table>
<tr>
<td>
<pre>
  <profile>mxf-OP1a-MPEG</profile>
  <tcoffset/>
  <completed>0</completed>
  <progress>0</progress>
  <startms>100345709</startms>
  <currentms>-1</currentms>
  <type/>
  <message>Copy Pending</message>
 </copyid>
 <result>0</result>
 <resultstring>returning requested
copy</resultstring>
</copy>
</pre>
</td>
<td>
<pre>
  "target": "d:\\record\\second_file"
 },
 "netxbaseip": "192.168.50.100",
 "result": "-1",
 "resultstring": "copy not found"
}
</pre>
</td>
</tr>
</table>

Will not remove or purge copies that have errors or are marked as complete.

Parameters:
target=<targetfile>

Because copy slots are re-used, it is possible to have more than one complete for the same group. If you are asking for the group specifically, you will need to ask for each complete until there are no more. If you are asking for the -1 all group, in the return the multiple completes for a single channel will all be there. Once a complete (or progress) has been requested, it is removed from our list and is no longer available.

3.3 Typical Proxy, Convert and PFR Session

This section has some typical proxy, convert and partial file restore workflows. The first scenario is a complete list. The subsequent scenarios assume the first few steps in the first scenario have already been accomplished.

3.3.1 Scenario 1 – full access

- The file (MXF, MOV, AVI, CINE, etc.) arrives at ingest.
- A “command=copy&profile=index” command is sent to index the original filename.
- A “command=copy&profile=mp4-h264” command is sent to make a proxy file with time code, multitrack audio, closed captions, metadata and proxy index filename.
- Two “command=copy&pisrc&pidst” commands are sent to create JPEG images for the source and proxy files.
- At this point, the proxy and main index can be stored in a real or near real time storage, and

the main file may be moved to long term storage, tape, cloud (google/s3) or other offline storage.

- The user uses the HTML5 player's time code (or other time code source) to set one or more In and Out points on the file that needs to be restored.
- A "command=copy&profile=wrap" is sent to access the bytes of the original file and create a new file of the same type, without any recompression of audio or video, at the target location.

3.3.2 Scenario 2 – tape restore

- This assumes the basic processing in Scenario 1 has been done
- A "command=getcopyinout" is sent with the absolute or time code based in and out points, and the index of source file it will come from
- This returns a series of one or more file names with start and end byte locations
- At this point, the controller restores those byte areas of the files to the name specified by the return
- Once the areas are restored, a "command=copy&profile=wrap" is sent along with the temp folder to create the new output filename

3.3.3 Scenario 3 – cloud restore

- This assumes the basic processing in Scenario 1 has been done
- If the index file is stored on cloud, it can be restored locally first, or read directly from the cloud (https, ftps, aws)
- If the main file is in Glacier, then a command will be sent to restore the section needed to S3 before the restore is done
- Once there is access to the file, or file part, the "command=copy&profile=wrap" can be called normally
- If the resource is in available cloud storage (e.g. not Glacier), then partial file restores may be done from the original file without indexing it first. This will cause more data to be read, but only the headers and tables necessary to find the audio/video/data the restoration needs

3.3.4 Scenario 4 – in line conversions

- This assumes the basic processing in Scenario 1 has been done
- For any restore scenario, the file restored can be a byte accurate re-wrap of the original into a new container, or it can be translated in process (on the fly) to any supported standard format. These formats include MXF Op1a, Op-Atom, P2, IMX, D11, IMF, MOV, Uncompressed and many other containers, with codecs including JPEG-2000, XDCam, MPEG-2, h.264, HEVC, AVCi 100/200, XAVC-S. XAVC, Long-G, TR-01, DV and many others

- The commands can also be used with or without index files to convert all or part of local clips to any of these formats

3.4 PFR File Best Practices

Different workflows require different ways of saving and restoring files, but there are some general rules that can make it easier, especially when working with tape or other non sequential storage systems. Media files can be roughly broken down into a few categories:

- Self contained, single files (MXF OP1a, MOV, AVI)
- Multiple stream files (AVI+WAV, Avid OPAtom, MOV QT Reference)
- Multiple files per stream (P2 MXF, XDCam MP4, Canon C300/700 MXF)
- Sequences (DPX, TGA, TIFF)

If you are restoring from reasonable speed, random access devices, then all these types can be simply stored and retrieved as is. If there are sequential access, speed or cost issues, then it makes more sense to make each of these as easily accessible as possible before storing them.

3.4.1 Self Contained

These can be indexed and stored directly.

3.4.2 Multiple Stream Files

These can also be indexed and stored directly, as the RTIN can point at one file per stream. Net-X-Code automatically finds and joins the parts of the streams if it is a supported file type like Avid OPAtom, MOV reference, or video file with rationally named audio. If they are stored this way, there will be one chunk from each file that needs to be restored to make the output file, as the media is in separate files. This is handled in the getinoutbytes return as a series of temp names and start/end byte ranges. If you prefer a single chunk, please follow the guidelines in the Multiple Files Per Stream section.

3.4.3 Multiple Files Per Stream

The RTIN cannot describe streams that have multiple parts per stream. These are normally broken up to get around older disk format restrictions, often at 2 or 4 gigabytes. To deal with these, they should be pre processed (wrapped) to a self contained file like MXF OP1a before they are stored to tape. Using Net-X-Code to wrap them will cause the original audio/video to be copied to the new MXF

without recompression and will automatically generate the RTIN as it is creating the new file. A proxy file can also optionally be created while rewrapping the file. This MXF now becomes the file you would restore from and that should be stored to tape.

Net-X-Code fully supports automatically joining the parts of most broadcast and post production files for playback and wrapping, including

- Panasonic P2
- IMF
- AS-02
- DCP
- Canon C300/700
- Sony XDCam MXF
- Sony XDCam MP4
- Avid OP-Atom
- Grass Valley K2 Server Format
- Multi card MXF capture

3.4.4 Avid OP-Atom – Special Case

Net-X-Code supports automatically joining Avid OP-Atom files and creating Avid OP-Atom files that can be directly dropped into the Avid bin for pickup on the next database refresh. If you are using this workflow, then then the Multiple Stream Files method works best. If you are importing files into Avid then you will not be able to use OP-Atom, as Avid cannot import even its own. It will want to see the same essence, but in an OP1a container. In this case, you should re-wrap the OP-Atom to OP1a before storing them. It is also possible to restore OP-Atom to OP1a using the “type mxf-op1a” when doing the restore, but if they are stored as OP-Atom you will still need to restore a chunk from each stream to PFR them.

3.4.5 Sequences

As sequences can be restored ‘per file’, they do not need to be indexed.

3.5 Command - set

COMMANDS: Require client address and group id or group keyname, as well as a command (start, stop, restart, add, remove). These commands (start, stop, restart, add and remove) cannot be combined with the SETTINGS commands listed below. To have a command applied to all available groups, use the 'group=all'.

(Note that result values returned in XML structure indicate that the command was identified and dispatched by the HTTP server, they do not reflect the success or failure of the command result)

3.5.1 start

Start a group recording

http://127.0.0.1:1080/netx?request=set&client=0&group=0&command=start

<request netxbaseip="192.168.50.100" type="command=start">
  <start success="1">success</start>
  <result>0</result>
  <resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "start": {
    "success": "1",
    "value": "success"
  },
  "type": "command=start"
}
```


3.5.2 startstream

Start a group/channel streaming

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&command=startstream

<request netxbaseip="192.168.50.100" type="command=startstream">
  <start success="1">success</start>
  <result>0</result>
  <resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "start": {
    "success": "1",
    "value": "success"
  },
  "type": "command=startstream"
}
```


3.5.3 stop

Stop a group record.

http://127.0.0.1:1080/netxjson?request=set&client=0&group=0&command=stop

<request netxbaseip="192.168.50.100"
type="command=start">
  <start success="1">success</start>
  <result>0</result>
  <resultstring>success</resultstring>
</request>

{
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "stop": {
    "success": "1",
    "value": "success"
  },
  "type": "command=stop"
}

3.5.4 restart

Restart a group and make all settings current.

You may supply the index of the group:

http://127.0.0.1:1080/netx?request=set&client=0&group=0&command=restart

<request netxbaseip="192.168.50.100"
type="command=start">
  <start success="1">success</start>
  <result>0</result>
  <resultstring>success</resultstring>
</request>

{
  "netxbaseip": "192.168.50.100",
  "restart": {
    "success": "1",
    "value": "success"
  },
  "result": "0",
  "resultstring": "success",
  "type": "command=restart"
}

Or the keyname of a group:

http://127.0.0.1:1080/netx?request=set&client=0&group=ab1&command=restart

Or -1 or all to restart all groups:

http://127.0.0.1:1080/netx?request=set&client=0&group=all&command=restart
http://127.0.0.1:1080/netx?request=set&client=0&group=-1&command=restart

Or -2 to restart all the groups as well as the client:

http://127.0.0.1:1080/netx?request=set&client=0&group=-2&command=restart

<request netxbaseip="192.168.50.100" type="command=restart">
  <restart success="1">success</restart>
  <result>0</result>
  <resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.50.100",
  "restart": {
    "success": "1",
    "value": "success"
  },
  "result": "0",
  "resultstring": "success",
  "type": "command=restart"
}
```


3.5.5 xmldisable

Disable XML sidecar generation for captured files

This command gets the state of XML sidecar generation

http://127.0.0.1:1080/netxjson?request=set&client=0&group=0&channel=0&xmldisable=1

<request netxbaseip="192.168.50.100" type="channel">
  <setting success="1">success</setting>
  <result>0</result>
  <resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "setting": {
    "success": "1",
    "value": "success"
  },
  "type": "channel"
}
```


If set to 0, no sidecar XML files will be generated. If set to one, standard Drastic XML sidecar files will be generated.

3.5.6 autostart

Autostart, if enabled, causes any stream that is lost due to Net-X-Cmd or a Net-X-Code stopping, to

be automatically restarted when they are restarted. There are two levels of autostart:

1. <client> autostart must be enabled for any kind of autostart to occur on a client (Net-X-Cmd) server. If this is disabled, the next level is ignored.
2. <group> if the client autostart is enabled, then the Net-X-Cmd will look at the group autostarts to determine if a group should be restarted when Net-X-Cmd or Net-X-Code gets closed. This allows the caller to set up a server that only restarts some channels automatically

To set the client/NetXCmd autostart, use

http://127.0.0.1:1080/netx?request=set&client=0&autostart=0

```
<request netxbaseip="192.168.50.100"
type="channel">
  <setting success="1">success</setting>
  <result>0</result>
  <resultstring>success</resultstring>
</request>
```

```
{
  "autostart": {
    "success": "1",
    "value": "success"
  },
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "type": ""
}
```

To set the group within a client, use

http://127.0.0.1:1080/netxjson?request=set&client=0&group=0&autostart=1

```
<request netxbaseip="192.168.50.100"
type="channel">
  <setting success="1">success</setting>
  <result>0</result>
  <resultstring>success</resultstring>
</request>
```

```
{
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "setting": {
    "success": "1",
    "value": "success"
  },
  "type": "channel"
}
```

Note: If setting autostart during group creation, the &parameter=value set needs to be included only once per group, it is not on a per channel basis. If not specified, the default value for the group is 1 (true). For each Net-X-Cmd instance that runs, the default value will also be true until it is changed by the user.

3.5.7 add

Add a new group (group # not used for this action). Like all commands, this will return as soon as the command has been validated, but likely before the group has actually been created. A request for the channel using the key= name can determine when the channel has actually been allocated. There are 3 ways to add a group:

- You must supply a unique key name and it must contain at least one alpha character that isn't -.
- 0, 5, 243, and -1 are all invalid key names.
- aa, 365b, and -5abc are all valid.

a. add an empty group:

http://127.0.0.1:1080/netx?request=set&client=0&command=add&key=groupname

<request netxbaseip="192.168.50.100"
type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>


```
{
  "add": {
    "success": "1",
    "value": "success"
  },
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "type": "command=add"
}
```


b. setting all channels with defined default values:

http://127.0.0.1:1080/netxjson?
request=set&client=0&command=add&key=groupname&channel=-
1&address=239.255.40.40&port=1234&type=.mov&protocol=UDP

<request netxbaseip="192.168.50.100"
type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "type": "command=copy"
}
```


c. setting one or all channels with specific values:

http://127.0.0.1:1080/netx?

request=set&client=0&command=add&key=groupname&channel=0&address=239.255.40.40&name=chan1&port=1111&type=.mp4&protocol=RTP&channel=1&name=chan2&port=1222&address=239.255.41.41&type=.ts&protocol=UDP&channel=2&name=chan2&port=1333&address=239.255.42.42&type=.mov&protocol=RTP

<request netxbaseip="192.168.50.100"
type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "success",
	"type": "command=copy"
	}

NOTICE: When adding groups with channel settings, the channel specified must exist before its settings. &channel=0&name=name will work, &name=name&channel=0 will not. However, the order of the channels does not matter.

3.5.8 remove

Remove a group. Please note, this will only work on one channel at a time. The 'group=all' is not supported for this command.

http://127.0.0.1:1080/netx?request=set&client=0&group=0&command=remove

<request netxbaseip="192.168.50.100"
type="command=remove">
<remove success="1">success</remove>
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "success",
	"type": "command=copy"
	}

3.5.9 starttc/endtc

Set the desired start/end time for the given channel(s). Any channel(s) must be inactive for the setting to take effect. Once the tc values have been set and the given channel(s) have started, then the tc values will no longer be valid for subsequent connections and must then be set again. The format for the tc value is simply 24 hour time of day with an optional date part. The two forms are:

##:##:##:## (e.g. 17:00:00:00 for 5pm)
##:##:##:##-dd-mm-yyyy (e.g. 07:00:00:00-02-01-2017 for 7AM 2nd of January, 2017)
Note: all digits must be filled in, including leading zeros. 8 digits for time code, 2 for day, 2 for month and 4 for year.

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&starttc=11:11:11:11&endtc=22:22:22:22

<request netxbaseip="192.168.50.100"
type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "success",
	"type": "command=copy"
	}

3.5.10 delete

Delete one or more files on the file system

http://127.0.0.1:1080/netx?request=set&client=0&command=delete&file=E:/Record/netx/copy/source/temp.txt

http://127.0.0.1:1080/netx?request=set&client=9&command=delete&file=E:/Record/netx/copy/source/temp1.txt&file=E:/Record/netx/copy/source/temp 2.txt

<request netxbaseip="192.168.50.100"
type="command=delete">
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "success",
	"type": "command=copy"
	}

3.5.11 Proxymode

Set the proxy creation mode for all created files from stream source, SDI recording and clipping

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&proxymode=1&proxyfiletype=197&proxycompressiontype=1635148593&proxydatarate=5000

<request netxbaseip="192.168.50.100"
type="channel">
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "success",
	"type": "command=copy"
	}

- 0 disables proxy generation
- 1 creates proxies when a file is written to

- 2 creates a proxy when a file is read from
- 3 always creates a proxy

3.5.12 proxyfiletype

Set the proxy file type

http://127.0.0.1:1080/netx?request=set&client=0&channel=0&group=1&proxymode=1&proxyfiletype=197&proxycompressiontype=1635148593&proxydatarate=5000

<request netxbaseip="192.168.50.100" type="channel">
  <result>0</result>
  <resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "type": "command=copy"
}
```


- **197** Standard MP4
- **210** Multirate HLS (Apple live video)
- **211** Multirate DASH (Fragmented MP4 live video)
- **201** MXF AS-02

3.5.13 proxycompression

Set the proxy compression. Currently must be h.264 = 1635148593

http://127.0.0.1:1080/netx?request=set&client=0&channel=0&group=0&proxymode=1&proxyfiletype=197&proxycompressiontype=1635148593&proxydatarate=5000

<request netxbaseip="192.168.50.100" type="channel">
  <result>0</result>
  <resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "type": "command=copy"
}
```

3.5.14 proxydatarate

Set the proxy datarate in kilobits per second (e.g. 5000 = 5 megabits)

```
http://127.0.0.1:1080/netx?request=set&client=0&channel=0&group=0&proxymode=1&proxyfiletype=197&proxycompressiontype=1635148593&proxydatarate=5000
```

<request netxbaseip="192.168.50.100" type="channel">
  <result>0</result>
  <resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "type": "command=copy"
}
```


3.5.15 tsenable

Enable secondary transport stream capture. If capturing an MXF, MP4, fMP4, MOV or other main file type, enabling this will cause a second capture of the raw transport stream, with associated rtin and m3u8, to be captured as well. Before enabling, the tsdir and tsfile should also be set.

```
http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&tsenable=1
```

<request netxbaseip="192.168.100.189" type="channel">
        <setting success="1">success</setting>
        <result>0</result>
        <resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.100.189",
  "result": "0",
  "resultstring": "success",
  "setting": {
    "success": "1",
    "value": "success"
  },
  "type": "channel"
}
```


3.5.16 tsdir

Set the directory where the secondary transport stream capture will be stored. The tsenable must be 1 for this to have effect.

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&tsdir=/tmp/capture

<request netxbaseip="192.168.100.189" type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.100.189",
  "result": "0",
  "resultstring": "success",
  "setting": {
    "success": "1",
    "value": "success"
  },
  "type": "channel"
}
```


3.5.17 tsfile

Set the file name for the secondary transport stream capture. The tsenable must be 1 for this to have effect.

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&tsfile=hls.ts

<request netxbaseip="192.168.100.189" type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.100.189",
  "result": "0",
  "resultstring": "success",
  "setting": {
    "success": "1",
    "value": "success"
  },
  "type": "channel"
}
```


3.6 SETTINGS

Requires client address, group id number of key name, channel, parameter and a value. Settings can now be grouped, so multiple settings to one channel can be sent as one command. The settings commands cannot, however, be combined with the commands above (start, stop, restart, add and remove). These would have to be sent as two separate commands.

- group – the group or key name

http://127.0.0.1:1080/netx?request=set&client=0&group=0&key=newgroupname

<request netxbaseip="192.168.100.189" type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.100.189",
	"result": "0",
	"resultstring": "success",
	"setting": {
	"success": "1",
	"value": "success"
	},
	"type": "channel"
	}

- name - File name

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=1&name=myname

<request netxbaseip="192.168.100.189" type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.100.189",
	"result": "0",
	"resultstring": "success",
	"setting": {
	"success": "1",
	"value": "success"
	},
	"type": "channel"
	}

- address - Source stream address

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=1&address=239.255.40.44

<request netxbaseip="192.168.100.189" type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.100.189",
	"result": "0",
	"resultstring": "success",
	"setting": {
	"success": "1",
	"value": "success"
	},
	"type": "channel"
	}

- port - Source stream port

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=1&port=5004

<request netxbaseip="192.168.100.189" type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.100.189",
  "result": "0",
  "resultstring": "success",
  "setting": {
    "success": "1",
    "value": "success"
  },
  "type": "channel"
}
```


- protocol - Network capture protocol: udp, rtp

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=1&protocol=rtp

<request netxbaseip="192.168.100.189" type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.100.189",
  "result": "0",
  "resultstring": "success",
  "setting": {
    "success": "1",
    "value": "success"
  },
  "type": "channel"
}
```


- directory - File directory

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=1&directory=/Volumes/capturedrive/current/

<request netxbaseip="192.168.100.189" type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.100.189",
  "result": "0",
  "resultstring": "success",
  "setting": {
    "success": "1",
    "value": "success"
  },
  "type": "channel"
}
```

- rectc – Time code recording method:
  - **0** Record every frame time code (with TC PID Only)
  - **1** Record every I frame, interpolate between (recommended) (with TC PID Only)
  - **2** Record the first time code, interpolate forward (with TC PID Only)
  - **3** Record time code converted from first PTS, interpolate forward
  - **4** Record time of day as time code, interpolate forward

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=1&rectc=1

<request netxbaseip="192.168.100.189" type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.100.189",
  "result": "0",
  "resultstring": "success",
  "setting": {
    "success": "1",
    "value": "success"
  },
  "type": "channel"
}
```


- type - File target type: .ts, .mp4, .fmp4, .mov, .ism, .mxf, .h264

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=1&type=.mp4

<request netxbaseip="192.168.100.189" type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.100.189",
  "result": "0",
  "resultstring": "success",
  "setting": {
    "success": "1",
    "value": "success"
  },
  "type": "channel"
}
```


- previewenabled – Enable/disable preview for channel(s). Can be set across all groups/channels. Can also use the “preview” parameter tag.

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=1&previewenabled=1

<request netxbaseip="192.168.100.189" type="channel">
<setting success="1">success</setting>


```
{
  "netxbaseip": "192.168.100.189",
  "result": "0",
```

<result>0</result>
<resultstring>success</resultstring>
</request>
	"resultstring": "success",
	"setting": {
	"success": "1",
	"value": "success"
	},
	"type": "channel"
	}

- threshold – Stream data rate threshold. Set the data rate alarm limit.

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=1&threshold=6291456

<request netxbaseip="192.168.100.189" type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.100.189",
	"result": "0",
	"resultstring": "success",
	"setting": {
	"success": "1",
	"value": "success"
	},
	"type": "channel"
	}

The settings commands can also be combined into one command, but settings commands cannot be mixed with the group commands above. The settings must be set, then the group command sent separately. For example:

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=1&name=myname&address=239.255.40.44&port=5004&protocol=rtp&directory=/Volumes/capturedrive/current/&type=.mp4

<request netxbaseip="192.168.100.189" type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.100.189",
	"result": "0",
	"resultstring": "success",
	"setting": {
	"success": "1",
	"value": "success"
	},
	"type": "channel"
	}

http://127.0.0.1:1080/netx?request=set&client=0&group=0&command=restart

<request netxbaseip="192.168.100.189"
type="channel">
  <setting success="1">success</setting>
  <result>0</result>
  <resultstring>success</resultstring>
</request>
	{
	  "netxbaseip": "192.168.100.189",
	  "result": "0",
	  "resultstring": "success",
	  "setting": {
	  "success": "1",
	  "value": "success"
	  },
	  "type": "channel"
	}

NOTE: if group is actively capturing from a stream, setting changes will not take effect until the group is restarted

NOTE: it is recommended to update all statuses for any client that has had a command performed on it.

3.7 Media File Scanner Command Set

Media File Scanner REST API provides many commands that allow you to scan directories.
If there are directories waiting to be scanned when Media File Scanner is launched, the command below is called right away:

3.7.1 Getting a List of Scanning Instances

getfirstscandir will return the very first directory scanned

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&getfirstscandir
```

```
<mediafilescanner>
      <firstscandir>D:\t3media\captions\</firstscandir>
      <connectionstatus>Connected IP 192.168.100.229</connectionstatus>
</mediafilescanner>
```

Let's say the directory returned from getfirstscandir is stored in a variable called directoryReceived. In order to keep checking for more directories, directoryReceived must be used in the next API call:

getnextscandir

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=D:\t3media\captions\&getnextscandir
```

```
<mediafilescanner>
      <nextscandir>m:\t3media\streams\</nextscandir>
</mediafilescanner>
```

Once all the **scandir** instances are exhausted, the **getnextscandir** will returned

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=m:\t3media\streams&getnextscandir
```

```
<mediafilescanner>
      <nextscandir>.</nextscandir>
</mediafilescanner>
```

The **scandir** element is necessary for certain commands as it will identify which directory path to perform specific actions on

getnextscandir will return the next directory scanned if there is one. If no directory is found then "." is returned

3.7.2 Adding new Media File Scanner (newscandir=)

In order to add a new Media File Scanner, three important pieces of information is required. The directory name, database name and the database management system must be included in the API call. The database management system can either be a value of 1 or 2 where 1 represents SQLite and 2 represents MySQL.

The command below will add a new Media File Scanner if SQLite (1) is selected:

http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&**newscandir**=directoryName&**setdatabase**=databaseName&**setdatabasemanagement**=dbManagementValue

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&newscandir=d:\record&setdatabase=d:\t3media\sqlite_test.db&setdatabasemanagement=1
```

```
<mediafilescanner>
      <newscandiradded>d:\record</newscandiradded>
</mediafilescanner>
```

Assume directoryName contains the actual directory path, databaseName contains the database path and dbManagementValue contains the value of which database management system to use

newscandir will add a new Media File Scanner to the database
setdatabase will set the path of the database so data can be written
setdatabasemanagement will set the type of database management system to user-defined

NOTE: The order of parameters in the API above is extremely important
Order of parameters:

1) **newscandir**
2) **setdatabase**
3) **setdatabasemanagement**

If MySQL (2) is selected, the user must provide username, password and port to access the MySQL

database
The command below will add a new Media File Scanner if MySQL (2) is selected:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**newscandir**=directoryName&**setdatabase**=databaseName&**setusername**=username&**setpassword**=password&**setport**=portVal&**setdatabasemanagement**=dbManagementValue

newscandir will add a new Media File Scanner to the database
setdatabase will set the path of the database so data can be written
setusername will set the username
setpassword will set the password
setport will set the port value
setdatabasemanagement will set the type of database management system to user-defined

NOTE: The order of parameters in the API call above is extremely important
Order of parameters:

1) **newscandir**
2) **setdatabase**
3) **setusername**
4) **setpassword**
5) **setport**
6) **setdatabasemanagement**

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&newscandir=d:\record&setdatabase=mysq_test&setusername=root&setpassword=root&setport=0setdatabasemanagement=1
```

```
<mediafilescanner>
      <newscandiradded>d:\record</newscandiradded>
</mediafilescanner>
```

3.7.3 Deleting An Instance

Any instance can be deleted. Please note, this will not delete the database or any generated files, but rather just the instance itself, and the saved values. There is no way to reverse this command.

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&scandir=directoryName&**delete**=1

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&delete=1
```

```
<mediafilescanner>
	<value>0</value>
</mediafilescanner>
```

3.7.4 Clearing Errors For An Instance

This command clears all the errors in an instance, making the clips eligible for processing on the next scan.

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&clearerrors
```

```
<mediafilescanner>
	<value>0</value>
</mediafilescanner>
```

3.7.5 Set/Get Directory Path or Database

In order to change the directory path two pieces of information are required. User must provide the current directory path being scanned and the new directory path to be scanned. Assume the current directory path being scanned is in a variable called curDirPath and the new directory path to be scanned is in a variable called newDirPath.

The command below will change the directory path of a MediaFileScanner:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=curDirPath&**setcurrentdir**=newDirPath

scandir is identifying which directory path to perform the task on
setcurrentdir will set the directory path to the value stored in newDirPath

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&delete=1
```

```
<mediafilescanner>
	<value>0</value>
</mediafilescanner>
```

The command below will get the current directory being scanned:
http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&scandir=curDirPath&getcurrentdir

scandir is identifying which directory path to perform the task on
getcurrentdir will get the current directory associated with the Media File Scanner being scanned which is stored in curDirPath

```
http://192.168.100.229:1080/netx?
request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getcurrentdir
```

```
<mediafilescanner>
      <currentdir>d:\record\</currentdir>
</mediafilescanner>
```

Assume curDirPath holds the directory path of the Media File Scanner and newDatabaseValue holds the new database path the user wants to change it to.

The command below will change the database of a specific Media File Scanner based on the directory path and the new database value:

http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=curDirPath&**setdatabase**= newDatabaseValue

scandir is identifying which directory path to perform the task on
setdatabase will change the path of the database being used

```
http://192.168.100.229:1080/netx?
request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setdatabase=record1.db
```

```
<mediafilescanner>
      <setdatabase>record1</setdatabase>
</mediafilescanner>
```

The command below will get the database path based on curDirPath:

http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=curDirPath&**getdatabase**

scandir is identifying which directory path to perform the task on
getdatabase will get the database associated with the Media File Scanner being scanned which is stored in curDirPath

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getdatabase
```

```
<mediafilescanner>
      <setdatabase>record1</setdatabase>
</mediafilescanner>
```

GetDBInfo will return all of the settings around the database, including **username=**, **password=**, **host=**, and **databasename-**.

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getbinfo
```

```
<mediafilescanner>
      <username>User</username>
      <password>Pass</password>
      <host>localhost</host>
      <databasename>roger</databasename>
</mediafilescanner>
```

3.7.6 Skipping Extensions or Paths

3.7.6.1 Skip Extensions

To skip files with certain extensions, use setskipextensions along with a string of extensions (.ext1.ext3.ext3)

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setskipextensions=.mxf.mov.avi
```

```
<mediafilescanner>
      <setskipextensions>.mxf.mov.avi</setskipextensions>
</mediafilescanner>
```

To retrieve the current extensions being skipped, use getskipextensions:

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:
```

\record&getskipextensions

```
<mediafilescanner>
      <getskipextensions>.mxf.mov.avi</getskipextensions>
</mediafilescanner>
```

3.7.6.2 Skip Strings (Paths)

To skip files with certain strings in their path, use setskipstringsdir along with a set of strings (C:\DontScan;DontScanAnyWithThisString;)

http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setskipstringsdir=str1;str2

```
<mediafilescanner>
      <setskipstringsdir>str1;str2;</setskipstringsdir>
</mediafilescanner>
```

To retrieve the current extensions being skipped, use getskipstringsdir:

http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getskipstringsdir

```
<mediafilescanner>
      <getskipstringsdir>str1;str2;</getskipstringsdir>
</mediafilescanner>
```

3.7.6.3 Skip Strings (Name)

To skip files with certain strings in their path, use setskipstringsname along with a set of string (scan;DontScanAnyWithThisString;_proxy;)

http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setskipstringsname=str1;str2

```
<mediafilescanner>
      <setskipstringsname>str1;str2;</setskipstringsname>
</mediafilescanner>
```

To retrieve the current extensions being skipped, use getskipstringsdir:

http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:

\record&getskipstringsname

```
<mediafilescanner>
      <getskipstringsname>str1;str2;</getskipstringsname>
</mediafilescanner>
```

3.7.7 Starting a Scan

If there are directories waiting to be scanned as soon as Media File Scanner is run, the scan will initially start. If a new Media File Scanner is added, no scanning process is run. When a new Media File Scanner is created, user has the ability to either Start, Stop, Pause or Clear this newly created Media File Scanner. Assume directoryName contains the actual directory path.

start will start the scanning process of the directoryName. The command below will start a scan of the directory being used:

http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&scandir=directoryName&
start

http://192.168.100.229:1080/netx?
request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:
\record&start

```
<mediafilescanner>
      <start>0</start>
</mediafilescanner>
```

stop will stop the scanning process of the directoryName. The command below will stop a scan of the directory being used:

http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&scandir=directoryName&
stop

http://192.168.100.229:1080/netx?
request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:
\record&stop

```
<mediafilescanner>
      <stop>8</stop>
</mediafilescanner>
```

pause will pause the scanning process of the directoryName. The command below will pause the scanning process of the directory being used:

http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&scandir=directoryName&
pause

```
http://192.168.100.229:1080/netx?
request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:
\record&clear
```

```
<mediafilescanner>
      <clear>0</clear>
</mediafilescanner>
```

clear will clear the scanning process completely of the directoryName. The command below will clear the scanning process of the directory being used:

http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&scandir=directoryName&
clear

3.7.8 Monitoring a Scan

While Media File Scanner is working, you can monitor the scan with **getstate**. Getstate will return a main state (notrunning, processing, paused and stopped) as well as a sub state when returning processing. The current returns are:

- state=notrunning
- state=processing(FindDirectories)
- state=processing(FindClips)
- state=processing(ProcessingClips)
- state=processing(ProcessingComplete)
- state=paused
- state=stopped
- state=unknown

Please note, the state can go from processing(ProcessingComplete), back to processing(FindDirectories) or processing(FindClips) if new directories or clips are added to the scanning directory after the initial scan is complete.

3.7.8.1 gettotaldirectories

Returns the total number of directories found

3.7.8.2 gettotalfiles

Returns the total number of files found

3.7.8.3 gettotalclips

Returns the total number of clips (anchors) found

3.7.8.4 getprocesseswaiting

Returns the total number of processes currently running/waiting

3.7.8.5 getoperation

Returns the current operation state. Operation states include:

- NotRunning (1)
- FindingDirectories (2)
- FindingClips (3)
- ProcessingClips (4)
- Complete (5)
- Paused (6)
- Resuming (7)
- Stopped (8)
- Unknown (9)

3.7.9 Media File Scanner Configure

When a Media File Scanner is created, the user has the ability to enable, disable, and set many properties through the configuration.

Assume **dirPath** contains the directory path to perform the action on

3.7.9.1 Max Translations

Set or get the maximum allowed simultaneous operations allowed from the media file scanner.

setmaxtranslations

getmaxtranslations

3.7.9.2 Max MD5 Translations

Set or get the maximum allows simultaneous MD5 operations in the media file scanner

getmaxmd5s
setmaxmd5s

3.7.9.3 Make MD5

Assume makeMD5Value stores the value to make an MD5. makeMD5Value can only be 1 or 0 (true or false). The command below will allow the user the to make an MD5:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setmakemd5**= makeXMPValue

scandir is identifying which directory path to perform the task on
setmakemd5 will set Make MD5 according to the directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setmakemd5=0
```

```
<mediafilescanner>
      <setmakemd5>0</setmakemd5>
</mediafilescanner>
```

The command below will get the value of make MD5 based on the directory path:
http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getmakemd5**

scandir is identifying which directory path to perform the task on
getmakemd5 will get the value of make MD5. This value will tell the user if an MD5 should be made or not

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getmakemd5
```

```
<mediafilescanner>
```

```
      <makemd5>false</makemd5>
</mediafilescanner>
```

3.7.9.4 Make Picon

Assume makePiconValue stores the value. makePiconValue can only be 1 or 0 (true or false). The command below will allow the user the to make a Picon:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setmakepicon**= makePiconValue

scandir is identifying which directory path to perform the task on
setmakepicon will set Make Picon according to the directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setmakepicon=1
```

```
<mediafilescanner>
      <setmakepicon>1</setmakepicon>
</mediafilescanner>
```

The command below will get the value of make Picon:
http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getmakepicon**

scandir is identifying which directory path to perform the task on
getmakepicon will get the value of make Picon. This value will tell the user if a Picon should be made or not

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getmakepicon
```

```
<mediafilescanner>
      <makepicon>true</makepicon>
</mediafilescanner>
```

Picon Directory

To set the directory for the picon files, setpicondirectory can be used along with setpiconflags.

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setpicondirectory=/mnt/piconstore/
```

```
<mediafilescanner>
      <setpicondirectory>/mnt/piconstore/</setpicondirectory>
</mediafilescanner>
```

To retrieve the current picon directory, use getpicondirectory:

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getpicondirectory
```

```
<mediafilescanner>
      <setpicondirectory>/mnt/piconstore/</setpicondirectory>
</mediafilescanner>
```

Picon Flags

Assume piconFlagsValue stores the value. The command below will allow the user to set the Picon Flags:

setpiconflags will set the picon flag's value to its associated directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setpiconflags=2
```

```
<mediafilescanner>
      <setpiconflags>2</setpiconflags>
</mediafilescanner>
```

Values include:

- 0 – use an absolute path
- 1 – place the picon with the source file
- 2 – place the picon in a flat path (all in one directory)
- 3 – place the picon in the same directory on a new volume

The command below will get the value of Picon Flags:

getpiconflags will get the value of the Picon Flags

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getpiconflags
```

```
<mediafilescanner>
      <getpiconflags>2</getpiconflags>
</mediafilescanner>
```

Picon Percent
Assume piconPercentValue stores the value. The command below will allow the user to set the Picon Percentage:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setpiconpercent**=piconPercentValue

scandir is identifying which directory path to perform the task on
setpiconpercent will set the Picon Percent value to its associated directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setpiconpercent=22
```

```
<mediafilescanner>
      <setpiconpercent>22</setpiconpercent>
</mediafilescanner>
```

The command below will get the value of Picon Percent:
http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getpiconpercent**

scandir is identifying which directory path to perform the task on
getpiconpercent will get the value of Picon Percent.

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getpiconpercent
```

```
<mediafilescanner>
      <piconpercent>2200</piconpercent>
</mediafilescanner>
```

Picon Frame
Assume piconFrameValue stores the value. The command below will allow the user to set the Picon Frame:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setpiconframe**=piconFrameValue

scandir is identifying which directory path to perform the task on
setpiconframe will set the Picon Frame value to its associated directory path

http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setpiconframe=4

```
<mediafilescanner>
      <value>0</value>
</mediafilescanner>
```

The command below will get the value of Picon Frame:
http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getpiconframe**

scandir is identifying which directory path to perform the task on
getpiconframe will get the value of Picon Frame.

http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getpiconframe

```
<mediafilescanner>
      <piconframe>0</piconframe>
</mediafilescanner>
```

Picon Width

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setpiconwidth**=640

scandir is identifying which directory path to perform the task on
setpiconwidth will set the picon width value to its associated directory path

http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setpiconwidth=640

```
<mediafilescanner>
      <value>0</value>
</mediafilescanner>
```

The command below will get the value of Picon Frame:
http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getpic**

onwidth

scandir is identifying which directory path to perform the task on
getpiconframe will get the value of Picon Frame.

```
http://192.168.100.229:1080/netx?
request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:
\record&getpiconwidth
```

```
<mediafilescanner>
      <piconframe>1920</piconframe>
</mediafilescanner>
```

Picon Height

http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setpiconheight**=480

scandir is identifying which directory path to perform the task on
setpiconheight will set the picon height value to its associated directory path

```
http://192.168.100.229:1080/netx?
request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:
\record&setpiconheight=480
```

```
<mediafilescanner>
      <value>0</value>
</mediafilescanner>
```

The command below will get the value of Picon height:
http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getpiconheight**

scandir is identifying which directory path to perform the task on
getpiconheight will get the value of Picon height.

```
http://192.168.100.229:1080/netx?
request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:
\record&getpiconheight
```

```
<mediafilescanner>
      <piconheight>1080</piconheight>
</mediafilescanner>
```

3.7.9.5 Make Proxy

Assume makeProxyValue stores the value. makeProxyValue can only have a value of 1 or 0 (true or false). The command below will allow the user to set the Make Proxy:

http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setmakeproxy**= makeProxyValue

scandir is identifying which directory path to perform the task on
setmakeproxy will set the Make Proxy value to its associated directory path

```
http://192.168.100.229:1080/netx?
request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:
\record&setmakeproxy=1
```

```
<mediafilescanner>
      <setmakeproxy>1</setmakeproxy>
</mediafilescanner>
```

The command below will get the value of Make Proxy:
http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getmakeproxy**

scandir is identifying which directory path to perform the task on
getmakeproxy will get the value of Make Proxy.

```
http://192.168.100.229:1080/netx?
request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:
\record&getmakeproxy
```

```
<mediafilescanner>
      <makeproxy>true</makeproxy>
</mediafilescanner>
```

Assume proxyModeValue stores the value. proxyModeValue can only have a value of either mp4, dash or hls. The command below will allow the user to set the Proxy Mode

http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setproxymode**= proxyModeValue

scandir is identifying which directory path to perform the task on

setproxymode will set the Proxy Mode value to its associated directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setproxymode=mp4
```

```
<mediafilescanner>
      <setproxymode>mp4</setproxymode>
</mediafilescanner>
```

The command below will get the value of Make Proxy:
http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getproxymode**

scandir is identifying which directory path to perform the task on
getproxymode will get the value of Proxy Mode.

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getproxymode
```

```
<mediafilescanner>
      <proxymode>mp4</proxymode>
</mediafilescanner>
```

Available modes are: mp4, hls, dash

Proxy Width
Assume proxyWidthValue stores the value. The command below will allow the user to set the Proxy Width:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setproxywidth**= proxyWidthValue

scandir is identifying which directory path to perform the task on
setproxywidth will set the proxy width to its associated directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setproxywidth=320
```

```
<mediafilescanner>
      <setproxywidth>320</setproxywidth>
</mediafilescanner>
```

The command below will get the value of Proxy Width:
http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getproxywidth**

scandir is identifying which directory path to perform the task on
getproxywidth will get the value of the Proxy Width.

```
http://192.168.100.229:1080/netx?
request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getproxywidth
```

```
<mediafilescanner>
      <getproxywidth>320</getproxywidth>
</mediafilescanner>
```

Proxy Height

Assume proxyHeightValue stores the value. The command below will allow the user to set the Proxy Height:

http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setproxyheight**= proxyHeightValue

scandir is identifying which directory path to perform the task on
setproxyheight will set the proxy height to its associated directory path

```
http://192.168.100.229:1080/netx?
request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setproxyheight=240
```

```
<mediafilescanner>
      <setproxyheight>240</setproxyheight>
</mediafilescanner>
```

The command below will get the value of Proxy Height:
http://127.0.0.1:1080/netx?
request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getproxyheight**

scandir is identifying which directory path to perform the task on
getproxyheight will get the value of the Proxy Height.

```
http://192.168.100.229:1080/netx?
```

request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getproxyheight

```
<mediafilescanner>
      <getproxyheight>240</getproxyheight>
</mediafilescanner>
```

Proxy Scale

Assume proxyScaleValue stores the value. The command below will allow the user to set the Proxy Scale:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setproxyscale**= proxyScaleValue

scandir is identifying which directory path to perform the task on
setproxyscale will set the Proxy Scale to its associated directory path

http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setproxyscale=1

```
<mediafilescanner>
      <setproxyscale>1</setproxyscale>
</mediafilescanner>
```

The command below will get the value of Proxy Scale:
http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getproxyscale**

scandir is identifying which directory path to perform the task on
getproxyscale will get the value of the Proxy Scale.

http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getproxyscale

```
<mediafilescanner>
      <getproxyscale>0</getproxyscale>
</mediafilescanner>
```

Proxy Compression

Assume proxyCompressionValue stores the value. The command below will allow the user to set the Proxy Compression:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setproxycompression**= proxyCompressionValue

scandir is identifying which directory path to perform the task on
setproxycompression will set the proxy compression to its associated directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setproxycompression=1635148593
```

```
<mediafilescanner>
      <setproxycompression>1635148593</setproxycompression>
</mediafilescanner>
```

The command below will get the value of Proxy Compression:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getproxycompression**

scandir is identifying which directory path to perform the task on
getproxycompression will get the value of the Proxy Compression.

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getproxycompression
```

```
<mediafilescanner>
      <getproxycompression>1635148593</getproxycompression>
</mediafilescanner>
```

Proxy Date Rate

Assume proxyDateRateValue stores the value. The command below will allow the user to set the Proxy Date Rate:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setproxydaterate**= proxyDateRateValue

scandir is identifying which directory path to perform the task on
setproxydaterate will set the Proxy Date Rate value to its associated directory path

http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setproxydaterate=5000

```
<mediafilescanner>
      <setproxydaterate>5000</setproxydaterate>
</mediafilescanner>
```

The command below will get the value of Proxy Date Rate:
http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getproxydaterate**

scandir is identifying which directory path to perform the task on
getproxydaterate will get the value of the Proxy Date Rate.

http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getproxydaterate

```
<mediafilescanner>
      <getproxydaterate>2500</getproxydaterate>
</mediafilescanner>
```

Proxy Audio Route

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setproxyaroute**=78123456

scandir is identifying which directory path to perform the task on
setproxyaroute will set the audio channels to route to its associated directory path

http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setproxyaroute=78123456

```
<mediafilescanner>
      <setproxydaterate>5000</setproxydaterate>
</mediafilescanner>
```

The command below will get the value of audio route:
http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getproxyaroute**

scandir is identifying which directory path to perform the task on

getproxyaroute will get the value of the audio routing.

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getproxyaroute
```

```
<mediafilescanner>
      <getproxyaroute>78123456</getproxyaroute>
</mediafilescanner>
```

Proxy Flags

Assume proxyFlagsValue stores the value. The command below will allow the user to set the Proxy Flags:

setproxyflags will set the proxy flag's value to its associated directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setproxyflags=2
```

```
<mediafilescanner>
      <setproxyflags>2</setproxyflags>
</mediafilescanner>
```

Values include:

- 0 – use an absolute path
- 1 – place the proxy with the source file
- 2 – place the proxy in a flat path (all in one directory)
- 3 – place the proxy in the same directory on a new volume

The command below will get the value of Proxy Flags:

getproxyflags will get the value of the Proxy Flags

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getproxyflags
```

```
<mediafilescanner>
      <getproxyflags>2</getproxyflags>
</mediafilescanner>
```

Proxy Cmd Flags

Proxy command flags are the same as the -flags parameters for NetXCopy. The command below will allow the user to set the Proxy Cmd Flags:

setproxycmdflags will set the proxy flag's value to its associated directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setproxycmdflags=stereo
```

```
<mediafilescanner>
      <setproxycmdflags>stereo</setproxycmdflags>
</mediafilescanner>
```

Please see the NetXCopy -flags parameter for available values.

The command below will get the value of Proxy Cmd Flags:

getproxycmdflags will get the value of the Proxy Cmd Flags

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getproxycmdflags
```

```
<mediafilescanner>
      <getproxycmdflags>stereo</getproxycmdflags>
</mediafilescanner>
```

Proxy Directory

To set the directory for the XMP files, setproxydirectory can be used along with setproxyflags.

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setproxydirectory=/mnt/proxystore/
```

```
<mediafilescanner>
      <setproxydirectory>/mnt/proxystore/</setproxydirectory>
</mediafilescanner>
```

To retrieve the current proxy director, use getproxydirectory:

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getproxydirectory
```

```
<mediafilescanner>
```

```
      <setproxydirectory>/mnt/proxystore/</setproxydirectory>
</mediafilescanner>
```

3.7.9.6 Set Mezzanine

Make Mezzanine

setmakemezzanine can be set to 1 or 0 (true or false). The command below will allow the user to enable or disable the creation of mezzanine files:

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setmakemezzanine=1
```

```
<mediafilescanner>
      <setmakemezzanine>1</setmakemezzanine>
</mediafilescanner>
```

To retrieve the current make mezzanine setting, use getmakemezzanine:

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getmakemezzanine
```

```
<mediafilescanner>
      <getmakemezzanine>1</getmakemezzanine>
</mediafilescanner>
```

Mezzanine Mode

To set the kind of file to create for the mezzanine file, setmezzaninemode can be used. This will set an output file type that matches the NetXCopy types:

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setmezzaninemode=mp4
```

```
<mediafilescanner>
      <setmezzaninemode>mov-proreshq</setmezzaninemode>
</mediafilescanner>
```

The available mezzanine types include:

wrap - re wrap file or part of a file
mp3-128kbps - Audio MP3 file
wave - Audio wave file

mov-YCbCr8Bit - QuickTime MOV 8 bit uncompressed YCbCr file
mov-dvcprohd - QuickTime MOV DVCPro HD (1080/720)
mp4-h264 - MPEG-4 h264 AAC Audio
mxf-xdcam-720p - True XDCam MXF 8 channel audio
mxf-dvcprohd-720p - MXF DVCPro HD 720p
mxf-xdcam-1080i - True XDCam MXF 1080i 8 channel audio
mxf-dvcprohd-1080i - MXF DVCPro HD 1080i 29/25 fps
mxf-OP1a-MPEG - OpenMXF XDCam MPEG-2 16 channel audio
mxf-OP1a-h264 - MXF h.264
mxf-OP1a-HDF - MXF MPEG-2 HDF Standard
mxf-as-11-sd-pal-dpp - MXF AS-11 SD PAL DPP
mxf-as-11-sd-ntsc-dpp - MXF AS-11 SD NTSC DPP
mxf-as-11-hd-dpp - MXF DPP AS-11 AVCi HD
mov-proreshq - QuickTime MOV ProRes HQ
mov-proreslt - QuickTime MOV ProRes LT
mov-prores422 - QuickTime MOV ProRes 422
mov-prores444 - QuickTime MOV ProRes 444(4)
scaledown2000k - MP4 264 960x540, 2mbs, AAC
scaledown500k - MP4 264 480x272, 0.5mbs, AAC
hd1080-5000kbs - MP4 HD 1080 with a target bitrate of 5 mbs
hd720-2500kbs - MP4 HD 720p with a target bitrate of 2.5 mbs
hd360-1250kbs - MP4 HD 360p with a target bitrate of 1.25 mbs
h264-7500kbs - MP4 Any resolution with a target bitrate of 7.5 mbs
Proxy-h264-5000kbs - MP4 high quality proxy for web
LBR-h264-10000kbs - Low bit rate, high quality local MP4
mxf-OP1a-JPEG2K - Samma style JPEG2000 YCbCr
mxf-AS-02-h264-10 - 10 bit 50 Mbs h.264 in AS-02 MXF
DASH-MP4-Multibitrate - Multi bitrate MP4s with DASH files
HLS-TS-Multibitrate - Multi bitrate TS streams with M3U8 files
MP4-MultiOutput - Multi MP4 with optional burn in files
TS-TR-01-JPEG-2000 - TR-01 JPEG-2000 transport stream
TS-MPEG2 - MPEG-2 4:2:0/passthrough transport stream
TS-h264 - h.264 4:2:0/passthrough transport stream
OP1a_HBR_50 - OP1a MXF h264 4:2:2 10 bit
mp4-XAVC-S_4_2_0 - MP4 Sony XAVC-S 4:2:0
mp4-XAVC-S_4_2_2 - MP4 Sony XAVC-S 4:2:2
aces - ACES image files
MXF-RDD-25 - MXF RDD-25 Proxy
dnxhd-mxf-720p - DNxHD 720p 50, 59, 60
dnxhd-mxf-1080p - DNxHD 1080p 25, 29
dnxhd-mxf-1080i - DNxHD 1080i 25, 29
dnxhr-mxf-10-hq - DNxHR High Quality 10 bit
dnxhr-mxf-8-hq - DNxHR High Quality 8 bit

dnxhr-mxf-sq - DNxHR Standard Quality
dnxhr-mxf-lq - DNxHR Low Quality
amt3-HQX_10 - AMT 3 DNx HQX 10 Bit
amt3-HiQuality - AMT 3 DNx High Quality
amt3-StandardQuality - AMT 3 DNx Standard Quality
amt3-LowQuality - AMT 3 DNx Low Quality
amt3-DNxHD36 - AMT 3 1080 DNxHD 36

To retrieve the current mezzanine mode, use getmezzaninemode:

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getmezzaninemode
```

```
<mediafilescanner>
      <getmezzaninemode>dnxhr-mxf-lq</getmezzaninemode>
</mediafilescanner>
```

Mezzanine Type

To set the sub type of file to create for the mezzanine file, setmezzaninetype can be used. This will set an output file type that matches the NetXCopy types:

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setmezzaninetype=mxf-amt
```

```
<mediafilescanner>
      <setmezzaninetype>mov-proreshq</setmezzaninetype>
</mediafilescanner>
```

- mxf-op1a - standard OP1a
- mxf-sonyhd - Sony XDCam compatible
- mxf-as02 - AS - 02 spec MXF
- mxf-avid - Avid OP-Atom (Drastic->bin)
- mxf-amt = Avid OP-Atom (Avid->aaf)
- mp4-fmp4 - Fragmented MP4 (normal MP4 if not set)
- mov - QuickTime MOV

To retrieve the current mezzanine mode, use getmezzaninetype:

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getmezzaninety[e
```

```
<mediafilescanner>
```

```
      <getmezzaninetype>mxf-op1a</getmezzaninetype>
</mediafilescanner>
```

Mezzanine Compression

Assume mezzanineCompressionValue stores the value. The command below will allow the user to set the Proxy Compression:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setpmezzaninecompression**= proxyCompressionValue

scandir is identifying which directory path to perform the task on
setmezzaninecompression will set the proxy compression to its associated directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setmezzaninecompression=1635148593
```

```
<mediafilescanner>
      <setmezzaninecompression>1635148593</setmezzaninecompression>
</mediafilescanner>
```

The command below will get the value of Mezzanine Compression:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getmezzaninecompression**

scandir is identifying which directory path to perform the task on
getmezzaninecompression will get the value of the Mezzanine Compression.

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getmezzaninecompression
```

```
<mediafilescanner>
      <getmezzaninecompression>1635148593</getmezzaninecompression>
</mediafilescanner>
```

Mezzanine Data Rate

The data rate value is in kilobits per second, so 5mbs would be 5000. The command below will allow the user the to set the Proxy data rate:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setpm**

ezzaninecompression= 50000

scandir is identifying which directory path to perform the task on
setmezzaninedatarate will set the proxy compression to its associated directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setmezzaninedatarate=5000
```

```
<mediafilescanner>
      <setmezzaninedatarate>5000</setmezzaninedatarate>
</mediafilescanner>
```

The command below will get the value of Mezzanine data rate:

http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getmezzaninedatarate**

scandir is identifying which directory path to perform the task on
getmezzaninedatarate will get the value of the Mezzanine data rate.

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getmezzaninedatarate
```

```
<mediafilescanner>
      <getmezzaninedatarate>5000</getmezzaninedatarate>
</mediafilescanner>
```

Mezzanine Width
“getmezzaninewidth”
“setmezzaninewidth”

Mezzanine Height
“getmezzanineheight”
“setmezzanineheight”

Mezzanine GOP Size
“getmezzaninegopsize”
“setmezzaninegopsize”

Mezzanine Audio Route
http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**setmez**

zaninearoute=78123456

scandir is identifying which directory path to perform the task on
setmezzaninearoute will set the audio channels to route to its associated directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setmezzaninearoute=78123456
```

```
<mediafilescanner>
      <setproxydaterate>5000</setproxydaterate>
</mediafilescanner>
```

The command below will get the value of audio route:
http://127.0.0.1:1080/netx?request=set&group=mfs&client=127.0.0.1&channel=0&scanner=0&mfscmd&**scandir**=dirPath&**getproxyaroute**

scandir is identifying which directory path to perform the task on
getproxyaroute will get the value of the audio routing.

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setmezzaninearoute
```

```
<mediafilescanner>
      <setmezzaninearoute>78123456</setmezzaninearoute>
</mediafilescanner>
```

Mezzanine Flags
Assume mezzanineFlagsValue stores the value. The command below will allow the user to set the Mezzanine Flags:

setmezzanineflags will set the mezzanine flag's value to its associated directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setmezzanineflags=2
```

```
<mediafilescanner>
      <setmezzanineflags>2</setmezzanineflags>
</mediafilescanner>
```

Values include:

- 0 – use an absolute path
- 1 – place the proxy with the source file
- 2 – place the proxy in a flat path (all in one directory)

- 3 – place the proxy in the same directory on a new volume

The command below will get the value of Mezzanine Flags:

getmezzanineflags will get the value of the Mezzanine Flags

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getmezzanineflags
```

```
<mediafilescanner>
      <getmezzanineflags>2</getmezzanineflags>
</mediafilescanner>
```

Mezzanine Directory

To set the directory for the mezzanine files, setmezzaninedirectory can be used along with setmezzanineflags.

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setmezzaninedirectory=/mnt/mezzaninestore/
```

```
<mediafilescanner>
      <setmezzaninedirectory>/mnt/mazzaninestore/</setmezzaninedirectory>
</mediafilescanner>
```

To retrieve the current mezzanine mode, use getmezzaninedirectory:

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getmezzaninedirectory
```

```
<mediafilescanner>
      <setmezzaninedirectory>/mnt/mezzaninestore/</setmezzaninedirectory>
</mediafilescanner>
```

Mezzanine NetXCopy Command Flags

Mezzanine command flags are the same as the -flags parameters for NetXCopy. The command below will allow the user to set the Mezzanine Cmd Flags:

setmezzaninecmdflags will set the proxy flag's value to its associated directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setmezzaninemdflags=stereo
```

```
<mediafilescanner>
      <setmezzaninecmdflags>stereo</setmezzaninecmdflags>
</mediafilescanner>
```

Please see the NetXCopy -flags parameter for available values.

The command below will get the value of Mezzanine Cmd Flags:

getmezzaninecmdflags will get the value of the Mezzanine Cmd Flags

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getmezzaninecmdflags
```

```
<mediafilescanner>
      <getmezzaninecmdflags>stereo</getmezzaninecmdflags>
</mediafilescanner>
```

3.7.9.7 Make XMP

Make XMP

setmakexmp can be set to 1 or 0 (true or false). The command below will allow the user to enable or disable the creation of XMP files:

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setmakexmp=1
```

```
<mediafilescanner>
      <setmakexmp>1</setmakexmp>
</mediafilescanner>
```

To retrieve the current make XMP setting, use getmakexmp:

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getmakexmp
```

```
<mediafilescanner>
      <getmakexmp>1</getmakexmp>
</mediafilescanner>
```

XMP Directory

To set the directory for the XMP files, setxmpdirectory can be used along with setxmpflags.

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setxmpdirectory=/mnt/editstore/
```

```
<mediafilescanner>
      <setxmpdirectory>/mnt/xmpstore/</setxmpdirectory>
</mediafilescanner>
```

To retrieve the current XMP mode, use getxmpdirectory:

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getxmpdirectory
```

```
<mediafilescanner>
      <setxmpdirectory>/mnt/xmpstore/</setxmpdirectory>
</mediafilescanner>
```

XMP Flags

Assume xmpFlagsValue stores the value. The command below will allow the user to set the XMP Flags:

setxmpflags will set the XMP flag's value to its associated directory path

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&setxmpflags=2
```

```
<mediafilescanner>
      <setxmpflags>2</setxmpflags>
</mediafilescanner>
```

Values include:

- 0 – use an absolute path
- 1 – place the XMP with the source file
- 2 – place the XMP in a flat path (all in one directory)
- 3 – place the XMP in the same directory on a new volume

The command below will get the value of XMP Flags:

getxmpflags will get the value of the XMP Flags

```
http://192.168.100.229:1080/netx?request=set&group=mfs&client=192.168.100.229&channel=0&scanner=0&mfscmd&scandir=d:\record&getxmpflags
```

```
<mediafilescanner>
      <getxmpflags>2</getxmpflags>
</mediafilescanner>
```

3.8 Creating an MD5

Assume baseIP is the IP address being used, srcPath is the source path and trgPath is the target path to copy the file over.
The command below will create a new MD5:
http://127.0.0.1:1080/netxjson?
request=set&client=baseIP&command=copy&profile=md5&source=srcPath&target=trgPath

client: baseIP is used to run on correct server
command: command will set correct operation to be completed
source: source is the path with MD5 object
target: target is the path to copy over the MD5 object to if needed

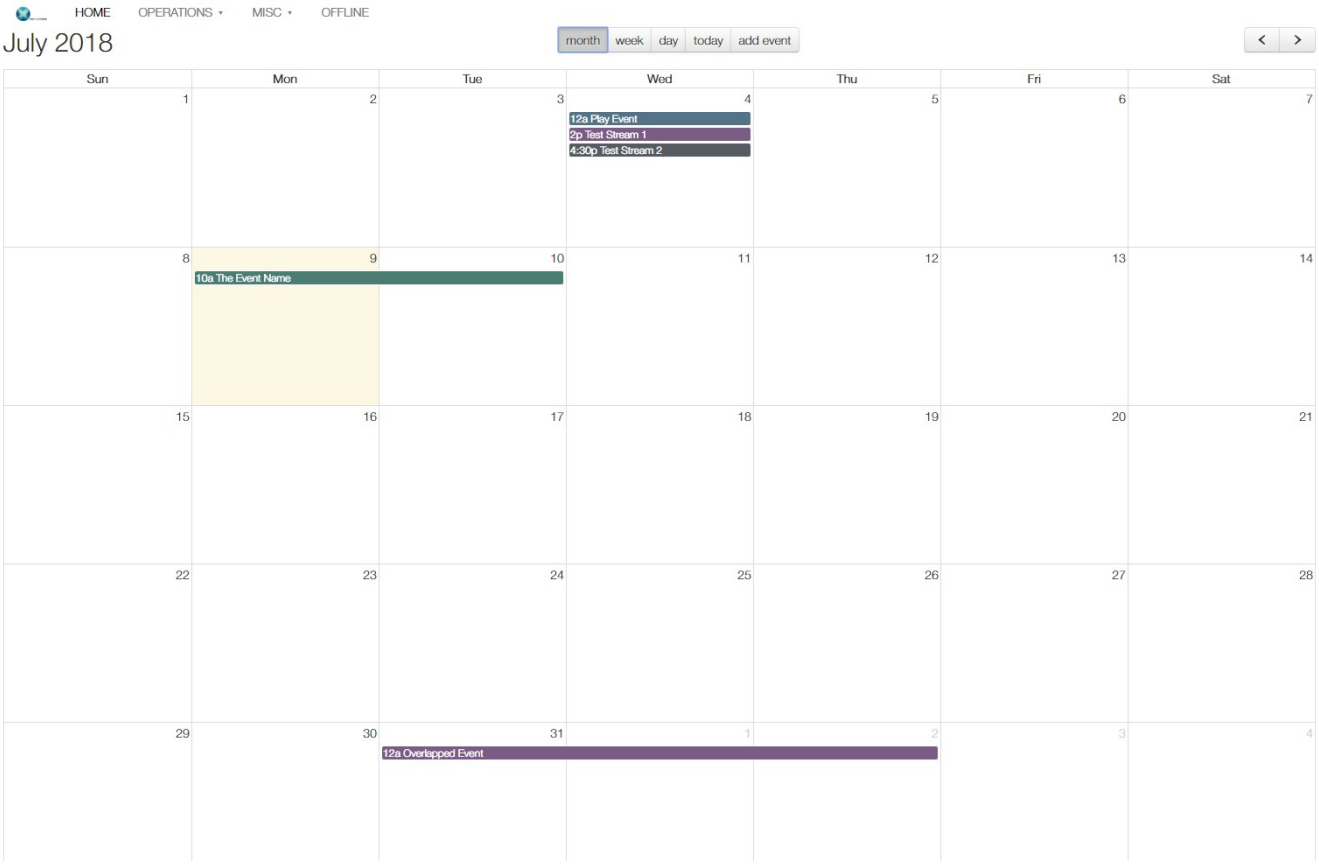
http://126.0.0.1:1080/netx?request=set&client=192.168.100.229&command=copy&source=M:\
t3media\Data\md5\Frame001048.dng&profile=md5

3.9 Scheduling

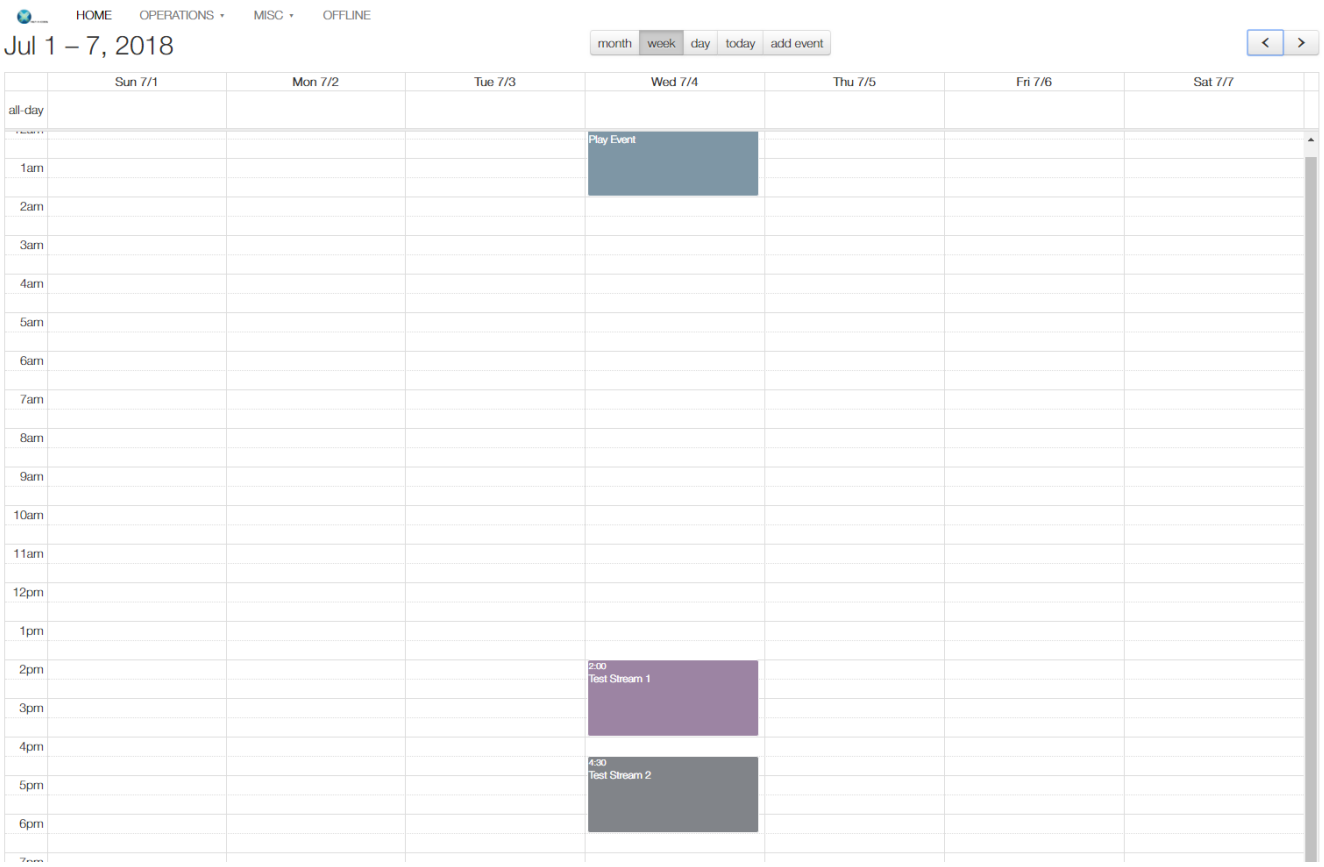
Net-X-Code includes a scheduling engine that allows you to set up events in the future that will be automatically triggered. These events may be set up in the HTML calendar provided with Net-X-Base, or can be set up/retrieved via the REST API specified here.

The calendar provides Month, Week, and Day Views. Clicking on the Month, Week, or Day tabs lets you switch between the views.

3.9.1 Month View

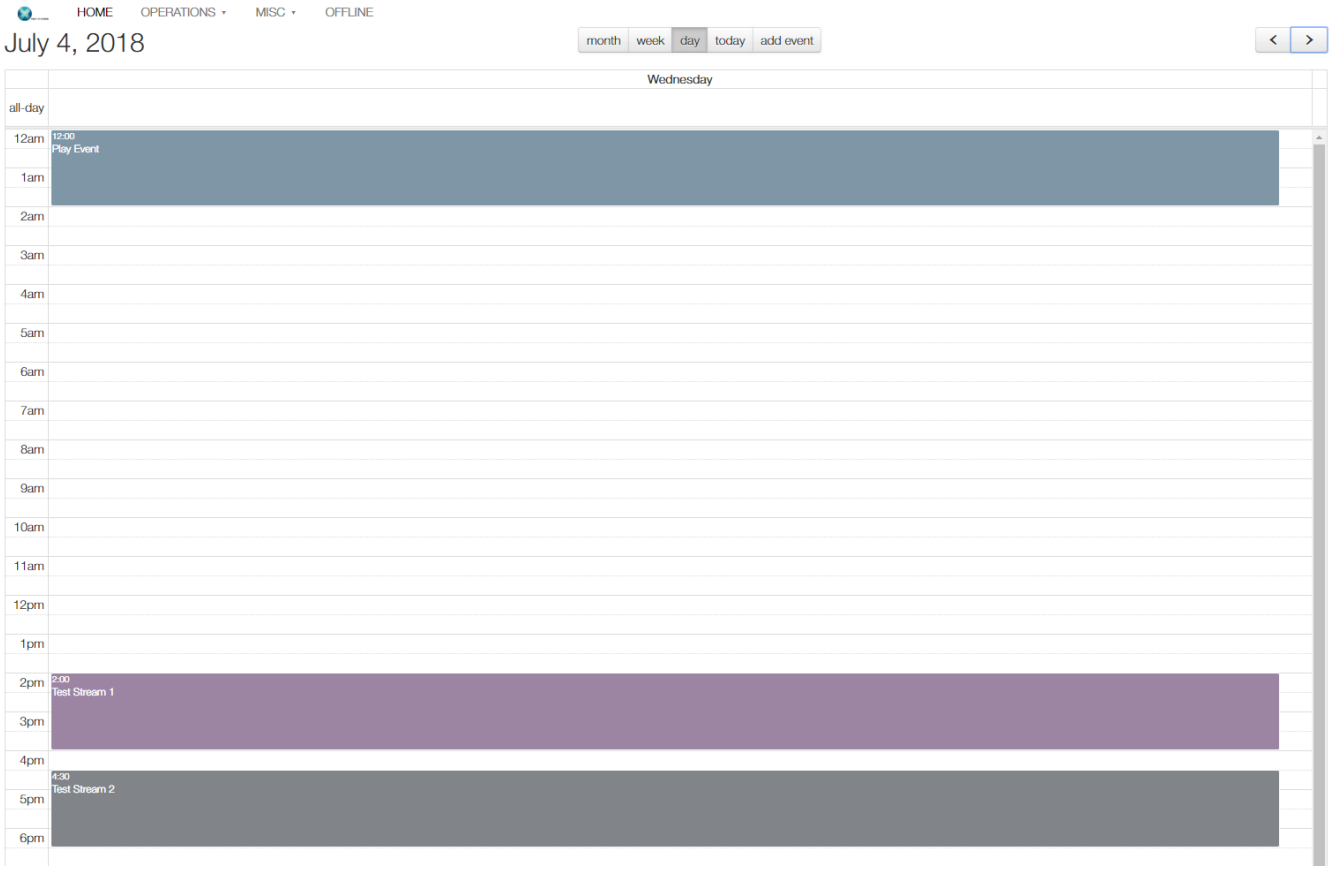


3.9.2 Week View



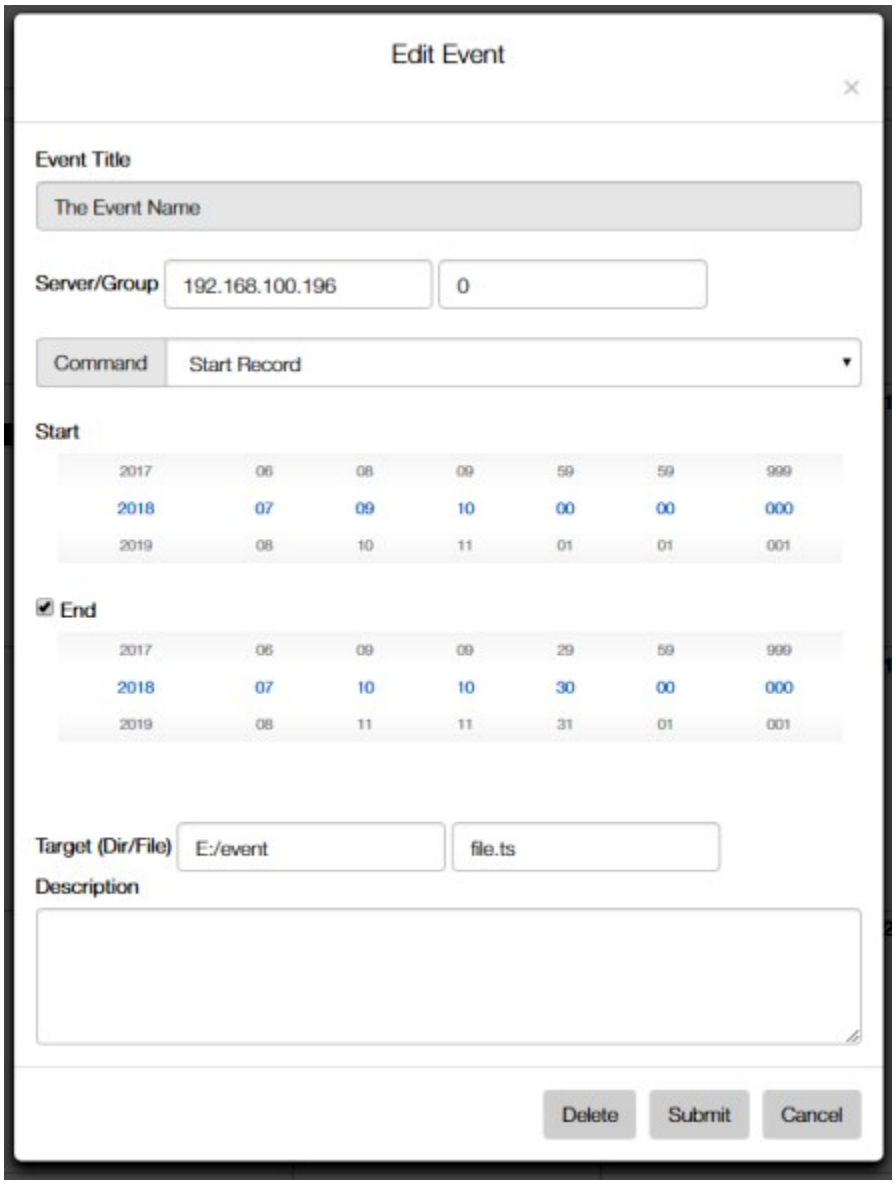
3.9.3 Day View

Pressing the Today tab returns the calendar to the present date.

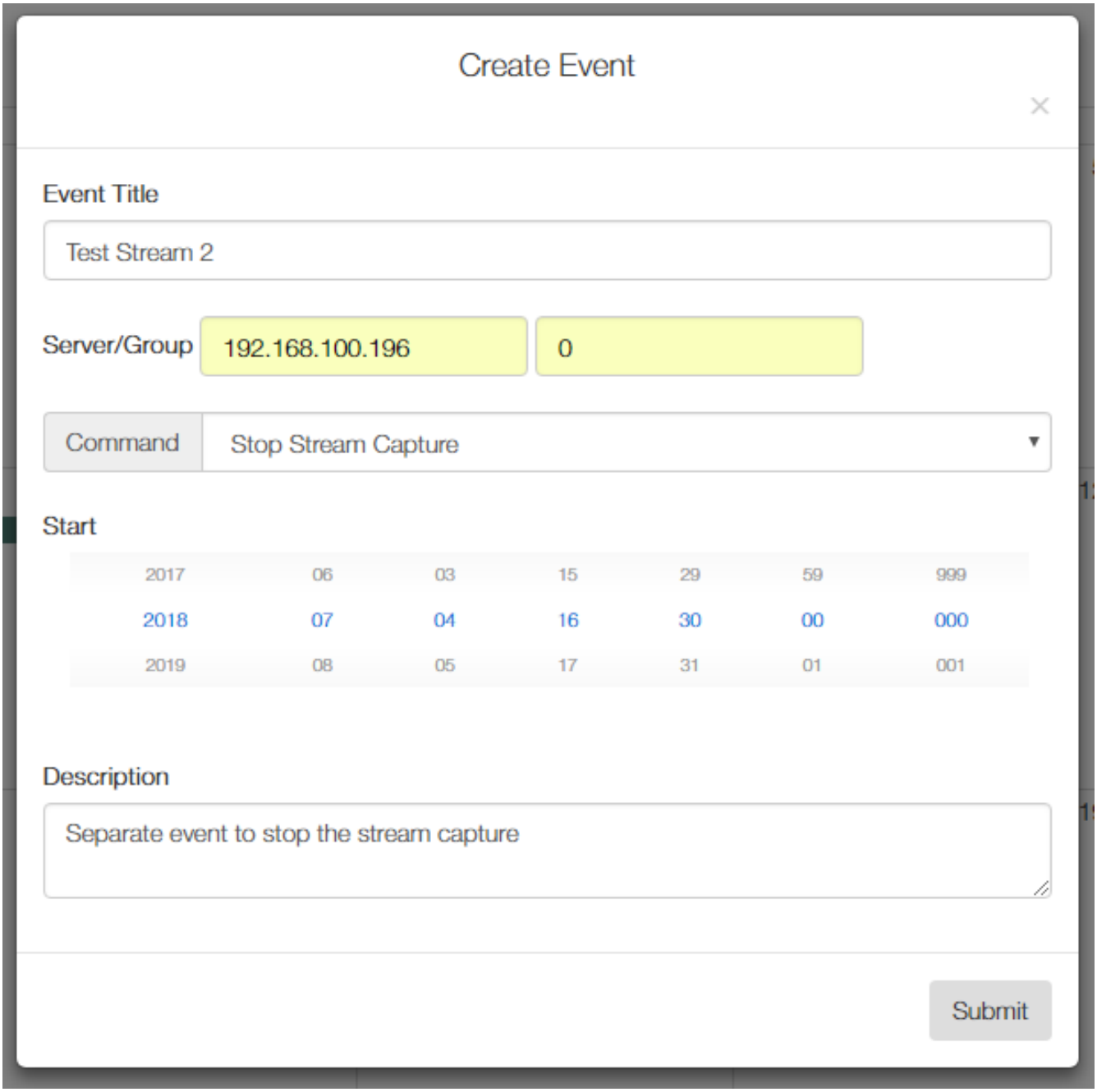


3.9.4 Create Event

Pressing the Create Event tab opens the event creation dialog. Specific events can include a start and end time, or a start and duration.

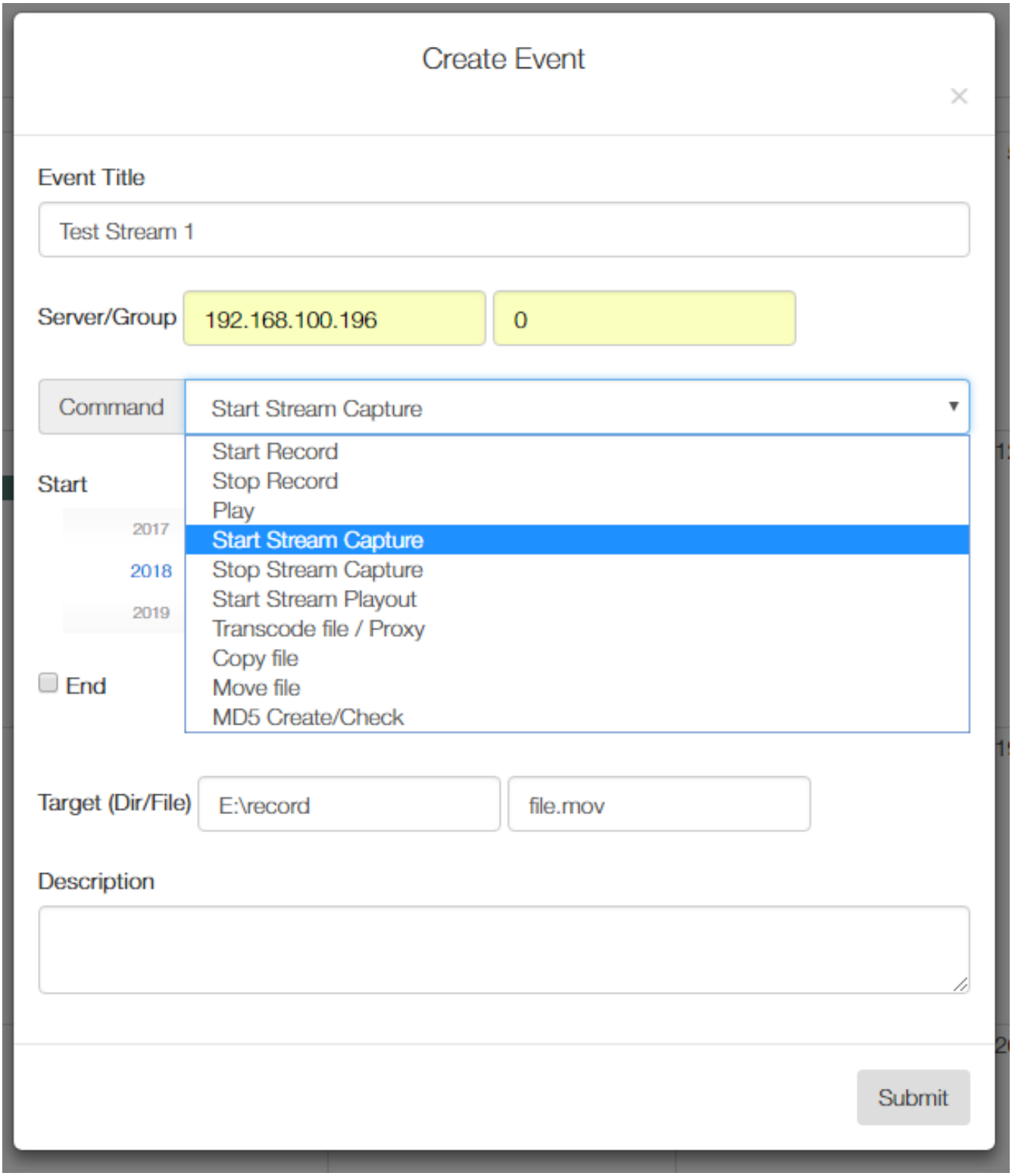


Or, the event can specify a start, and a separate event can be entered to end the process (such as a capture).



The event title can be entered, and the server and group specified.

There is a Command pulldown menu to set which command is used in the event.



The supported event types include:

- 0 - Start Record (SDI)
- 1 - Stop Record (SDI)
- 2 - Play (SDI)
- 3 - Start Stream Capture (IP)
- 4 - Stop Stream Capture (IP)
- 5 - Start Stream Playout (IP)
- 6 - Transcode File/Proxy (File)
- 7 - Copy (File)
- 8 - Move (File)
- 9 - MD5 Create/Check (File)

A scheduled event can contain the following parts:

- Server IP Address (server)
- Group Number or IP (group)
- Start Time/Date (start)
- End Time/Date (end)
- Type (type)
- Description (desc)
- Extended values depending on event (json)

The JSON format can include the following types

- Source directory (srcdir)
- Source file (srcfile)
- Target directory (trgdir)
- Target file (trgfile)
- Play mode (mode)
- Conversion profile (profile)
- MD5 directory (MD5Dir)
- MD4 file (MD5File)

3.9.5 Generate a new

```
http://127.0.0.1:1080/netxevent?
request=add&title=<thetitle>&server=<serverip>&group=#&start=YYYY-MM-DDTHH-mm-
ss&end=YYYY-MM-DDTHH-mm-ss&type=<type>&desc=<description>&json=<extended
parameters>
```

For example, to add an event

http://127.0.0.1:1080/netxevent?request=add&title=NewEVENT&server=192.168.50.100&group=0&start=2019-02-14T12-30-00&end=2019-02-14T14-30-00&type=0&desc=ATestEventToREcord

```
{
    "result": 1,
    "resultstring": "event added"
}
```

To update an existing event, use the same command but append the event's id number at the end (&id=#)

http://127.0.0.1:1080/netxevent?request=add&title=<thetitle>&server=<serverip>&group=#&start=YYYY-MM-DDTHH-mm-ss&end=YYYY-MM-DDTHH-mm-ss&type=<type>&desc=<description>&json=<extended parameters>&id=<#>

```
{
    "result": 1,
    "resultstring": "event updated"
}
```

3.9.6 Get a list of events

http://127.0.0.1:1080/netxevent?request=get&start=YYYY-MM-DDTHH-mm-ss&end=YYYY-MM-DDTHH-mm-ss
will return a JSON formatted string like this

```
{
    "events": [
        {
            "desc": "",
            "extra": "",
            "group": "0",
            "id": 1,
            "jsonInfo":
"{\"params\":[{\"name\":\"trgdir\",\"value\":\"E:/record\"},
{\"name\":\"trgfile\",\"value\":\"file.mov\"}]}",
            "result": "pending",
            "server": "192.168.100.196",
            "title": "Test Stream 1",
            "type": "3",
```

```
            "uid": "{66042049-b746-4f7f-9945-a441180f1cd7}"
        },
        {
            "desc": "Separate event to stop the stream capture",
            "extra": "",
            "group": "0",
            "id": 2,
            "jsonInfo": "",
            "result": "pending",
            "server": "192.168.100.196",
            "title": "Test Stream 2",
            "type": "4",
            "uid": "{e34001b7-4b56-4cc0-91c8-2e015dc6dcc2}"
        }
    ],
    "result": 0,
    "resultstring": "returning events"
}
```


3.10 QC/QA Processing

Net-X-Code supports up to 100 audio/video Quality Control/Quality Analysis sessions via set commands. These commands will take the IDs between -200 and -299.

3.10.1 To compare two files

To compare two files for basic video frame rate, audio channels, length of media, time code, closed captioning and metadata. This would be used to quickly check a proxy file generated from a source file, where the audio/video compression and size may be different, but the basic parameters should be the same. The progress value will run from 0 to 10000 while the file(s) are processed. The completed element will be set to one once the process has competed and the file result and resultString will be available at that point.

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&compare&file1=X:/media/james/qcfail/SOurce/ProRes422.mov&file2=X:/media/james/qcfail/Target/ProRes422.mp4


```
<request netxbaseip="192.168.100.189"
type="channel">
        <result>0</result>
        <resultstring>success</resultstring>
</request>
```

```
{
  "netxbaseip": "192.168.100.189",
  "result": "0",
  "resultstring": "success",
  "type": "channel"
```

	}

The compare will run, and then you can get the results of the comparison by calling

http://127.0.0.1:1080/netxjson?request=get&client=0&group=0&channel=0&compare&file1=X:/media/james/qcfail/SOurce/ProRes422.mov&file2=X:/media/james/qcfail/Target/ProRes422.mp4

<request>
<client/>
<command>compare</command>
<file>X:/media/james/qcfail/SOurce/ProRes422.mov</file>
<file2>X:/media/james/qcfail/Target/ProRes422.mp4</file2>
<progress>10000</progress>
<completed>1</completed>
<result>-109</result>
<resultstring>success</resultstring>
<xmlresult>
<ACKNOWLEDGE>
<DETAILS>
<FILE>
<FILENAME1>X:/media/james/qcfail/SOurce/ProRes422.mov</FILENAME1>
<FILENAME2>X:/media/james/qcfail/Target/ProRes422.mp4</FILENAME2>
<RESULT>-109</RESULT><COMMENT>ERROR system total length different (840 ! 300) </COMMENT>
<SRCPATH>X:/media/james/qcfail/SOurce/ProRes422.mov</SRCPATH>
<SRC2PATH>X:/media/james/qcfail/Target/ProRes422.mp4</SRC2PATH>
<ACKPATH>.ack</ACKPATH>
<TIME>Tue May 07 14:12:55 2019</TIME>
</FILE>
</DETAILS>
</ACKNOWLEDGE>
</xmlresult>
</request>

{
"client": "",
"command": "compare",
"completed": "1",
"file": "X:/media/james/qcfail/SOurce/ProRes422.mov",
"file2": "X:/media/james/qcfail/Target/ProRes422.mp4",
"progress": "10000",
"result": "-109",
"resultstring": "success",
"xmlresult": ""
}

3.10.2 To analyze the audio and video in a file

To do a single ended/no reference analysis, call the analyze command with the source file and optional database location. If the database is not specified, then it will be placed next to the source file with the same base name. Currently available analysis profiles include:

- psnr
- ssim
- ms-ssim

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&analyze&file1=X:/media/james/qcfail/SOurce/ProRes422.mov&qcfile=/mnt/storage/source.mxf.qc.db

<request netxbaseip="192.168.50.100" type="channel">
<result>0</result>
<resultstring>success</resultstring>
</request>


```
{
  "netxbaseip": "192.168.50.100",
  "result": "0",
  "resultstring": "success",
  "type": "command=copy"
}
```


While the analysis is running, the current progress can be retrieved with:

http://127.0.0.1:1080/netx?request=get&client=0&group=0&channel=0&**analyze**&profile=psnr&sourcefile=/mnt/storage/source.mxf&qcfile=/mnt/storage/source.mxf.qc.db

<request>
<client/>
<command>analyze</command>
<file>/mnt/storage/source.mxf</file>
<result>-1</result>
<resultstring>file not found</resultstring>
</request>


```
{
  "client": "",
  "command": "analyze",
  "file": "/mnt/storage/source.mxf",
  "result": "-1",
  "resultstring": "file not found"
}
```


Once the analysis is finished, the get will return a completed=1 and the location of the output file.

http://127.0.0.1:1080/netx?request=get&client=0&group=0&channel=0&**analyze**&profile=psnr&sourcefile=/mnt/storage/source.mxf&qcfile=/mnt/storage/source.mxf.qc.db

<request>
<client/>
<command>analyze</command>
<file>/mnt/storage/source.mxf</file>
<result>-1</result>


```
{
  "client": "",
  "command": "analyze",
  "file": "/mnt/storage/source.mxf",
  "result": "-1",
```

<resultstring>file not found</resultstring> </request>	"resultstring": "file not found" }

3.10.3 To validate a file against a pre-made profile

To validate a file against a pre-made profile, use the validate command. Validate requires the source file and the profile that you want to check it against. Please note: the profile must be available on all the possible Net-X-Code nodes that may do the QC process. Alternately, the client= can be used to force the validation to a particular Net-X-Code server.

http://127.0.0.1:1080/netx?request=get&client=0&group=0&channel=0&validate&file1=M:\t3media\Streams\MXF\Sony%20XDCam%20HD422\1920x1080_5994i_XDCAM_sample_20080312\1920x1080_5994i_XDCAM_sample_20080312\Clip\C0002.MXF&profile=XDCam%201080i60%2050Mbs

<request>
<client/>
<command>validate</command>
<file>
M:\t3media\Streams\MXF\Sony XDCam HD422\1920x1080_5994i_XDCAM_sample_20080312\1920x1080_5994i_XDCAM_sample_20080312\Clip\C0002.MXF
</file>
<result>-1</result>
<resultstring>file not found</resultstring>
</request>
	{
	"client": "",
	"command": "validate",
	"file": "M:\\t3media\\Streams\\MXF\\Sony XDCam HD422\\1920x1080_5994i_XDCAM_sample_20080312\\1920x1080_5994i_XDCAM_sample_20080312\\Clip\\C0002.MXF",
	"result": "-1",
	"resultstring": "file not found"
	}

3.10.4 To compare a file against another file for basic parameters

(for a proxy from a high res for example)

http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&**compare**&profile=basic&file1=/mnt/storage/source.mxf&file2=/mnt/proxy/source.mxf.mp4

<request netxbaseip="192.168.50.100" type="channel">
<setting success="1">success</setting>
<result>0</result>
<resultstring>success</resultstring>
</request>
	{
	"netxbaseip": "192.168.50.100",
	"result": "0",
	"resultstring": "success",
	"type": "command=copy"
	}

3.11Watch Folders

Up to 100 watch folders can be set up to trigger other Net-X-Code processes. These commands will take the IDs between -300 and -400. The primary watchprofile types are

- analyze – do a full reference analysis of two files (psnr, ssim, etc)
- qcfile – check a file against a specific profile, or another file

```
http://127.0.0.1:1080/netx?
request=set&client=0&group=0&channel=0&watchstart=&watchfolder=E:/watch/
source&watchprofile=analyze&watchfolder=E:/watch/target
```

```
<request netxbaseip="192.168.50.100" type="channel">
  <setting success="1">success</setting>
  <result>0</result>
  <resultstring>success</resultstring>
</request>
```

```
http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&watchid=-
300&watchcommand=start
```

```
<request netxbaseip="192.168.50.100" type="channel">
  <setting success="1">success</setting>
  <result>0</result>
  <resultstring>success</resultstring>
</request>
```

```
http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&watchid=-
300&watchcommand=stop
```

```
<request netxbaseip="192.168.50.100" type="channel">
  <setting success="1">success</setting>
  <result>0</result>
  <resultstring>success</resultstring>
</request>
```

3.12 Command - callstate

REQUIREMENTS: The client address and group id or group keyname as well as the command. If the command pertains to a channel, the channel index, parameter and value are required.

The "callstate" request is used to determine if a previous set command or set channel value has been processed by the system. There is a timeout involved with this functionality for performance concerns. The structure of the "callstate" command is the same as the set request for channel parameters, and very similar to the set request for a command. A typical cycle will look like this:

- issue command - Start a group recording

```
http://127.0.0.1:1080/netx?request=set&client=0&group=0&command=start
```

```
<request netxbaseip="192.168.50.100" type="command=start">
  <start success="1">success</start>
  <result>0</result>
  <resultstring>success</resultstring>
</request>
```

The response from the http server states that the request was received and issued to the system.

- callstate (command) - Query the system to see if it has been processed

```
http://127.0.0.1:1080/netx?request=callstate&client=&group=0&start=
```

The response from the http server shows a result code of -1 if the command does not exist in memory or if it has timed out. Otherwise the result will either be 0 if it has not been processed yet, or 1 if it has.

```
<client address="192.168.100.189">
        <result>0</result>
        <resultstring>call state returned 0</resultstring>
        <group index="0">
                <param>start</param>
                <result>0</result>
                <code>1</code>
                <description>ok</description>
        </group>
</client>
```

- issue set parameter value - Set name of a channel

```
http://127.0.0.1:1080/netx?request=set&client=0&group=0&channel=0&name=Channel1
```

```
<request netxbaseip="192.168.50.100" type="channel">
  <setting success="1">success</setting>
```

```
  <result>0</result>
  <resultstring>success</resultstring>
</request>
```

- callstate (channel param) - Query the system to see if it has been processed

```
http://127.0.0.1:1080/netx?request=callstate&client=0&group=0&channel=0&name=Channel1
```

```
<client address="192.168.100.229">
  <group index="0">
    <channel index="0">
      <param>name</param>
      <value>Channel1</value>
      <result>-1</result>
      <code>-1</code>
      <description>unspecified error</description>
    </channel>
  </group>
</client>
```

NOTE: Once a command or set parameter request has been issued, it is recorded in memory, it will become stale after 10 seconds if not set to 1 by the system indicating that it has been handled. Once it has been set to 1 however, it will live for an additional 10 seconds before being discarded. The exception is if the system detects that a request has failed, then the expire time will be set to 2 minutes from when the error was detected.

At any time the system can be queried for errors by making the following call:

```
http://127.0.0.1:1080/netx?request=callstate&client=0&group=0&errors=all
```

(The client need not be specified for this)

```
http://127.0.0.1:1080/netx?request=callstate&errors=all
```

The response will either be a statement that there are not any errors, or a list of requests that failed with an error count. Keep in mind that if an error was detected and it occurred longer than 2 minutes prior to the error check, it will not be reported.

Response with errors:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<errors count="2">
```

```
  <error index="0">
    <client address="192.168.100.109">
      <group index="0">
        <channel index="0">
          <param>name</param>
          <value>Channel1</value>
          <result>0</result>
          <code>-1</code> <-- recorded error
          <description>unable to set channel parameter</description
        </channel>
      </group>
    </client>
  </error>
  <error index="1">
    <client address="192.168.100.109">
      <group index="0">
        <channel index="0">
          <param>address</param>
          <value>239.40.40.41</value>
          <result>0</result>
          <code>-1</code> <-- recorded error
          <description>unable to set channel parameter</description>
        </channel>
      </group>
    </client>
  </error>
</errors>
```

Response without errors:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<errors count="0" />
```

NOTE: If an error occurs or if the command has not been processed and it expires, it will be written to a log file called "NetXCmdErrs.log" in the netxlog folder found in the user's home directory.

3.13 Command – close/restart

To close or restart a server on a client, use close or restart command with that client's IP address:

```
http://127.0.0.1:1080/netx?request=close&client=0
```

<request>
<result>0</result>
<resultstring>close - shutting down netxbase server</resultstring>
</request>
	{
	"result": "0",
	"resultstring": "close - shutting down netxbase server"
	}

http://127.0.0.1:1080/netx?request=restart&client=0

<request>
<result>0</result>
<resultstring>restart - restarting netxbase server</resultstring>
</request>
	{
	"result": "0",
	"resultstring": "restart - restarting netxbase server"
	}

3.14 Command – Thumbnail/JPEG/Picons

Net-X-Base also has the ability to create thumbnail/picon images for frames of video within a recorded or recording file, as well as return images for the current capture as a confidence monitor. The three main types of generation are: live confidence monitor, image on disk creation, image return as mime/jpeg for HTML. This feature is enabled or disabled with the 'previewenabled' command.

3.14.1 Live Confidence Monitoring

For any recording stream, a request can be made for the last I frame captured as a JPEG thumbnail. This can be returned as an XML with an embedded base64 JPEG image, or as a mime/jpeg for use directly in an <IMG> tag in a web page.

3.14.2 preview

http://127.0.0.1:1080/netx?request=get&client=0&group=0&channel=0&value=preview

<client netxbaseip="192.168.100.189" address="192.168.100.189" version="v7.0.0.38" groups="1" autostart="0">
<group index="0" key="newgroupname" version="v5.0.0.252" autostart="1">
<channel index="0"/>
</group>
<result>0</result>
	{
	"address": "192.168.100.189",
	"autostart": "0",
	"group0": {
	"autostart": "1",
	"channel": "",
	"index": "0",
	"key": "newgroupname",

<resultstring>group returned</resultstring> </client>	"version": "v5.0.0.252" }, "groups": "1", "netxbaseip": "192.168.100.189", "result": "0", "resultstring": "group returned", "version": "v7.0.0.38" }

The data returned is base 64 encoded JPEG data located in the following XML/JSON structure (in the example above the … is the data truncated).

3.14.3 mpreview

Getting a preview JPEG returned as a mime/jpeg suitable for an <IMG> tag.

```
http://127.0.0.1:1080/netx?request=get&client=0&group=0&channel=0&value=mpreview
```

Returns an actual JPEG image

The data returned is raw JPEG, marked in the HTML header as mime/jpeg. Note, this command does not support 'netxjson'.

3.14.4 Create JPEGs On Disk

Net-X-Copy can be used to create a JPEG image from a file (live or prerecorded) at any valid frame to any available location, from the Net-X-Code server used. Please note, if making images from recording streams you must use the associated RTIN file and not the main file of the record. Once the record is complete, either may be used, but the RTIN will be faster.

3.14.5 Picon

Create a JPEG picon to a target dir using the source name

```
http://127.0.0.1:1080/netx?request=picon&pisrc=d:/Record/second_file.rtin&pidst=d:/Record/
second_file.picon.jpg&pisize=10&piframe=25
```

Returns an actual JPEG image.

Create a JPEG picon to a target name. This command is not supported in 'netxjson'.

```
http://127.0.0.1:1080/netx?request=picon&pisrc=E:/Record/netx/copy/target/
dtl001.mov&pidst=E:/Record/netx/copy/target/dtl001.picon.jpg&pisize=10&piframe=25
```

To create a series of JPEG files from a file, with a distance between frames, the piskip parameter can be used. This command will create a numbered jpeg icon every 60 frames, starting at frame 150. This is useful for creating HTML scroll previews.

```
http://127.0.0.1:1080/netx?request=picon&pisrc=\mnt\Record\netx\target\dtl001.mov&pidst=\
mnt\Record\netx\proxy\/dt.jpg&width=180&piframe=150&piskip=60
```

3.14.6 Return A JPEG From A File

To get a JPEG suitable to display on a web page as an <IMG>, a call is made directly to the Net-X-Base. Net-X-Base must handle this request, as the picon has to be generated for the return from the call. As such, if you are doing a lot of generation, a separate instance of Net-X-Base should be set up, so that it does not interfere with the other operations Net-X-Base is handling for capture and clipping.

3.14.7 Picon

Create a JPEG picon and return it as a mime/jpeg (Note, this command is not supported with netxjson).

```
http://127.0.0.1:1080/netx?request=picon&pisrc=d:/Record/
amberfin_0000572516.rtin&pidst=&pisize=10&piframe=250
```

NBA
WEDNESDAY

3.15 NetXSDI – MediaCMD

Start NetXSdi.exe
http://127.0.0.1:1080/netx?request=set&client=0&group=sdi&sdistart&channel=0

Stop NetXSdi.exe
http://127.0.0.1:1080/netx?request=set&client=0&group=sdi&sdistop&channel=0

Get status:
http://127.0.0.1:1080/netx?request=set&client=0&group=sdi&sdistatus&channel=0

Process MediaCmd
http://127.0.0.1:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&play

Also there are:

- sdipicon
- sdinextclip
- sdiclipinfo
- sdiedlstate
- sdiedlinfo
- sdinextdirentry
- sdimakedir
- sdifileinfo
- sdierrormsg
- sdierrorlogsave

OLD:
http://localhost/VVWXMLMediaCmd?Play&speed=65520

REPLACE:
http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&play&speed=65520

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&

3.15.1 SetCurChannel

Use 'setchannel=

Sets the channel to which all subsequent commands will be sent. This command does not exist in the DLL interface as the channel is sent on a per command basis.

3.15.2 Play

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Play

Play at normal speed.

3.15.3 PlayAtSpeed

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Play &speed=lVVWSpeed

Play at a particular VVW speed. VVW speeds use a base play speed of 65520. This means that play = 65520, reverse play = -65520, four times play = 262080, half play speed = 32760. Percentage play speeds may be converted to VVW speeds using the PercentageToVVWSpeed() function. For Speed calculations please see GetSpeed() below.

Returns 0 if successful, else an error code.

3.15.4 PlayFromTo

http://localhost:1080/netx? request=set&client=0&group=sdi&sdicmd&channel=0&Play&start=lFrom &end=lTo

Play from a frame to another frame. As with editing systems, the ‘from’ point is included and will be displayed but the 'to' point is NOT included and will not be displayed. This means that the last frame displayed will be lFrom – 1. The deferred flag allows PlayFromTos to be stacked so that they will play back to back. The deferred flag in the status return should be false before another deferred command is added.

Returns 0 if successful, else an error code.

3.15.5 LoadClip

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Pause &ClipID=szClipname

Clip Mode Only. Load a clip into the channel and display the lStartFrame.

Returns 0 if successful, else an error code.

3.15.6 PlayClip

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Play&ClipID=szClipname&Flags=Deferred

Clip Mode Only. Play the entire clip specified by clip name. If the deferred flag is true, clip playback will only occur once the currently playing clip has finished. If there is no currently playing clip, playback will occur immediately.

Returns 0 if successful, else an error code.

3.15.7 PlayClipFromTo

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Play&start=lFrom&end=lTo&ClipID=szClipname

Clip Mode Only. Play the specified portion of the clip specified by clip name. If the deferred flag is true, clip playback will only occur once the currently playing clip has finished. If there is no clip currently playing, playback will occur immediately.

Returns 0 if successful, else an error code.

3.15.8 FastForward

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Play&speed=655200

Set the channel into its fastest possible forward motion state.

Returns 0 if successful, else an error code.

3.15.9 FastRewind

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Play

&speed=-655200

Set the channel into its fastest possible reverse motion state.

Returns 0 if successful, else an error code.

3.15.10 Pause

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Pause

Stop playback and display the current frame.

Returns 0 if successful, else an error code.

3.15.11 Seek

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Pause&position=lFrame

Seek to a particular frame and display it to the user. This call will return before the seek is complete. Once the Position return in the status reaches the lFrame, the seek is complete.

Returns 0 if successful, else an error code.

3.15.12 SeekRelative

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Pause&position=lFrameOffset

Seek a certain number of frames from the current position. Positive offsets imply forward direction, negative offsets imply reverse.

3.15.13 Stop

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Stop

Stop the output of the controlled channel and display the input video (not supported on all devices). On unsupported devices stop will be the same as a pause.

Returns 0 if successful, else an error code.

3.15.14 Record

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Record

Start the channel recording. In clip mode a default clip name will be used with a duration set to infinity. The record will stop on any transport command or at the point that the disk is full.

Returns 0 if successful, else an error code.

3.15.15 RecordFromTo

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Record&start=lStart&end=lEnd

Record from a frame value to a frame value. As with editing systems, the 'from' point is included and will be recorded but the to point is NOT included and will not be recorded. This means that the last frame recorded will be lFrom – 1.

Returns 0 if successful, else an error code.

3.15.16 RecordStop (prepare record)

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&RecStop

Clip Mode Only. Set the clip name and length of time to record in frames. The record will not actually start until Record() is called. If the lDuration is set to –1 the record will continue until Stop() is called or the channel runs out of space.

Returns 0 if successful, else an error code.

3.15.17 SetRecordPresets

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Record&videochannels=lVidEdit&audiochannels=lAudEdit**&infochannels=lInfEdit**

Set the channels to record. Using –1 values implies that the Preset should be set to all available channels. Record Presets will remain set until the user changes them.

Returns 0 if successful, else an error code.

3.15.18 Eject

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Eject

Eject the current media if it is removable. Normally only used with VTRs.

Returns 0 if successful, else an error code.

3.15.19 Transfer

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Transfer
&channel=lTargetChannel&position=lPosition
&start=lStart&end=lEnd&videochannels=lVidEdit
&audiochannels=lAudEdit&infochannels=lInfEdit
&Flags=Invert

Transfer media from one channel to another. Only supported by VTR channels. Currently only implemented for VTR to internal channels or internal channels to VTR channels. To record media from a VTR, the fToTape should be false, to record media onto a VTR the fToTape should be true. The start and end point are from the playback device. The edit will occur at the current time code location on the recorder.

Returns 0 if successful, else an error code.

3.15.20 Update Status

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&channel=0&VVWXMLGetStatus?

Retrieve the current status from the controlled device. The status is automatically updated by the interface, but this call ensures that the status is current when you are checking it.

Returns 0 if successful, else an error code.

VVWXMLGetStatus returns XML with a MediaCmd root element, for example:

```
<?xml version="1.0" ?>
  <MediaCmd>
  <!--  Drastic MEDIACMD xml structure version 1,0 -->
  <CmdID Value="-98238205" />
  <StructSize Value="336" />
  <Channel Value="-1" />
  <Cmd Value="1" UseClipID="1">Pause</Cmd>
  <Speed Value="0">0</Speed>
  <CmdAlt Value="2083947" TimeMs="1" />
  <Position Value="102" TcType="non-drop-frame" UsingFrameCount="1">00:00:03:12</Position>
  <Start Value="0" TcType="non-drop-frame" UsingFrameCount="1">00:00:00:00</Start>
  <End Value="2592000" TcType="non-drop-frame" UsingFrameCount="1">24:00:00:00</End>
  <ClipID>::VTR_TC</ClipID>
</MediaCmd>
```

3.15.21 GetState

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&VVWXMLGetStatus?

Returns the current state

- ctStop 0 // Stop all action
- ctPause 1 // Pause, Seek
- ctPlay 2 // Play at specified speed (includes pause)
- ctRecord 3 // Record at specified speed
- ctRecStop 4 // Stop ready for recording
- ctEject 5 // Eject the current media
- ctError 17 // An error has occurred
- ctAbort 19 // Abort any queued commands

XML: See <MediaCmd> root element, <Cmd> sub-element (value)

3.15.22 GetSpeed

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&VVWXMLGetStatus?

Returns the current VVW speed if the cfUseSpeed flag is set, otherwise pause or full play speed. VVW speeds are based on 65520 as the play speed. To translate to decimal number where 1.0 represents play, use the following formula:

D1Speed = ((double)VVWSpeed / 65520.0)

For percentages, where 100.0 represents play speed, use the following formula:

Dpercent = (((double)VVWSpeed * 100.0) / 65520.0)

= ((double)VVWSpeed / 655.2)

XML: See <MediaCmd> root element, <Speed> sub-element

Typical VVW speeds (note speeds are linear):

Pause	0%	0
Play	100%	65520
Half Play	50%	32760
Reverse Play	-100%	-65520
Reverse Double Play	-200%	-131040
10 Time Forward Play	1000%	655200
Max Forward Play	90000%	5896800
Max Reverse Play	-90000%	-5896800

3.15.23 GetPosition

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&VVWXMLGetStatus?

Returns the current position if the cfUsePosition flag is set, otherwise invalid.

XML: See <MediaCmd> root element, <Position> sub-element (value)

3.15.24 GetLastMS

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&VVWXMLGetStatus?

Returns the millisecond time the last status occurred (time of the last vertical blank).

XML: See <MediaCmd> root element, <CmdAlt> sub-element

3.15.25 GetStart

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&channel=0&VVWXMLGetStatus?

Returns the current start or in point if the cfUseStart flag is set.

XML: See <MediaCmd> root element, <Start> sub-element

3.15.26 GetEnd

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&channel=0&VVWXMLGetStatus?

Returns the current end point or out point if cfUseEnd is set.

XML: See <MediaCmd> root element, <End> sub-element

3.15.27 GetClipName

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&channel=0&VVWXMLGetStatus?

Only supported in clip Mode. Returns the current clip name, if any. For Direct access, the memory must be at least 9 bytes long (8 character bytes + NULL) and is always ANSI.

XML: See <MediaCmd> root element, <CmdID> sub-element

3.15.28 GetCurTC

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&channel=0&VVWXMLGetStatus?

Returns the current time code as a string (e.g. “00:01:00:00”). For Direct access, the memory must always be at least 15 bytes long (14 byte time code plus id + NULL) and is always ANSI.

XML: See <MediaCmd> root element, <Position> sub-element (text)

3.15.29 GetCurState

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&channel=0&VVWXMLGetStatus?

Returns the current state as a string (e.g. “Play”). For Direct access, the memory must always be at least 15 bytes long (14 byte state + NULL) and is always ANSI.

XML: See <MediaCmd> root element, <Cmd> sub-element (text)

3.15.30 GetNextClip

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&channel=0&VVWXMLNextClip?

Clip Mode Only. Returns the next clip identifier. To get the first clip, szLastClip should be an empty string. Once the last clip available has been returned, GetNextClip will return an error or NULL for Unix/DLL access. Please note: For Direct access, the sz8CharLastClipCurClip memory area is used for the new clip. The previous clip name is therefore lost and the memory is not allocated by the VVW.

Returns 0 if successful, else an error code.

VVWXMLNextClip returns XML with a ClipInfo root element, for example:

```
<?xml version="1.0" ?>
<ClipInfo>
  <!--  Drastic ClipInfo xml structure version 1,0 -->
  <ClipID>::Test</ClipID>
  <FileName>::Test</FileName>
  <Start Value="0" TcType="non-drop-frame">00:00:00:00</Start>
  <End Value="0" TcType="non-drop-frame">02:00:00:00</End>
</ClipInfo>
```

3.15.31 GetClipInfo

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&channel=0&VVWXMLClipInfo?

Returns the basic information from szClip. The information is located in lStart, lEnd, lVidEdit, lAudEdit and szFileName as the in point, out point, number of video channels, number of audio channels, and

the file name respectively.

Returns 0 if successful, else an error code.

XML: returns <ClipInfo> root element, <ClipID>, <FileName>, <Start>, <End> sub elements

3.15.32 EDLGetEdit

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&position=0**&videochannels=0&audiochannels=0&infochannels=0**

Returns an edit line from the VTR space of an internal channel. The function will continue to return the next edit in the time code space until the last edit is returned, after which an error will be returned. To reset to the start of the EDL use EDLResetToStart.

Returns 0 if successful else an Error code.

VVWXMLEDLInfo returns XML with a <MediaCmd> root element, for example:

```
 <?xml version="1.0" ?>
 <MediaCmd>
<!--  Drastic MEDIACMD xml structure version 1,0 -->
<CmdID Value="-98238205" />
<StructSize Value="336" />
<Channel Value="0" />
<Cmd Value="14" UseClipID="1">GetValue</Cmd>
<VideoChannels Value="1" />
<AudioChannels Value="0" />
<InfoChannels Value="0" />
<CmdAlt Value="93" />
<Position Value="5" TcType="non-drop-frame">00:00:00:05</Position>
<Start Value="0" TcType="non-drop-frame">00:00:00:00</Start>
<End Value="5" TcType="non-drop-frame">00:00:00:05</End>
<FileName>V:\Drastic Base Media\avi_er001_720x486_YUY2.avi</FileName>
 </MediaCmd>
```

3.15.33 Insert

http://localhost:1080/netx?

request=set&client=0&group=sdi&sdicmd&channel=0&Insert&ClipID=szClipName&position=lPosition&start=lStart&end=lEnd&videochannels=lVidEdit&audiochannels=lAudEdit&infochannels=lInfEdit&Flags=Ripple

Internal Channels Only. Do not use yet.

3.15.34 Blank

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Blank&ClipID=szClipName&position=lPosition&start=lStart&end=lEnd&videochannels=lVidEdit&audiochannels=lAudEdit&infochannels=lInfEdit&Flags=Ripple

Internal Channels Only. Do not use yet.

3.15.35 Delete

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Delete&ClipID=szClipName&position=lPosition&start=lStart&end=lEnd&videochannels=lVidEdit&audiochannels=lAudEdit&infochannels=lInfEdit&Flags=Ripple

Internal Channels Only. Do not use yet.

3.15.36 Trim

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&Trim&position=lPosition&start=lStartOffset&end=lEndOffset&videochannels=lVidEdit&audiochannels=lAudEdit&infochannels=lInfEdit&Flags=Ripple

Internal Channels Only. Do not use yet.

3.15.37 ValueSupported

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&ValueSupported&cmdalt=valuetype&position=lValueType

Returns the supported attributes of a get/set value (gsClipMode, gsTcSource, etc) or –1 for not supported.

3.15.38 GetValue

http://localhost:1080/netx?
request=set&group=sdi&sdicmd&client=0&getvalue&position=0&cmdalt=audwavepeakrms&audiochannels=0xc&videochannels=0&flags=-1

Returns the current setting for a get/set value.

Sample: get the RMS Peak of audio channels ½
http://localhost:1080/netx?
request=set&group=sdi&sdicmd&client=0&getvalue&position=0&cmdalt=audwavepeakrms&audiochannels=0x3&videochannels=0&flags=-1

3.15.39 SetValue

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&SetValue&cmdalt=valuetype**&position=lSetting**

Sets the get/set value to setting.

3.15.40 GetValue/SetValue Metadata

The Get and Set Values for metadata are split into two major groups: Integers and Strings. To get/set metadata, the string will include a “getvalue” or “setvalue”. The “cmdalt=metadata” must be included to indicate you are working with a metadata element. The metadata string, or equivalent integer, must be set to the “position=”. There are also some metadata elements that are read only, starting with VideoWidth to the end of the list.

Available metadata elements include:

3.15.40.1 Strings

```
//! see VVWINFO::szFileName
FileName,
//! see VVWINFO::szNativeLocator
NativeLocator,
//! see VVWINFO::szUniversalName
UniversalName,
//! see VVWINFO::szIP
IP,
//! see VVWINFO::szSourceLocator
SourceLocator,

//! see VVWINFO::szChannel
Channel,
//! see VVWINFO::szChannelName
ChannelName,
//! see VVWINFO::szChannelDescription
ChannelDescription,
//! see VVWINFO::szTitle
Title,
//! see VVWINFO::szSubject
Subject,
//! see VVWINFO::szCategory
Category,                   // <-- 10
//! see VVWINFO::szKeywords
Keywords,
//! see VVWINFO::szRatings
Ratings,
//! see VVWINFO::szComments
Comments,
//! see VVWINFO::szOwner
Owner,
//! see VVWINFO::szEditor
Editor,
//! see VVWINFO::szSupplier
Supplier,
//! see VVWINFO::szSource
Source,
//! see VVWINFO::szProject
Project,
//! see VVWINFO::szStatus
Status,
//! see VVWINFO::szAuthor
Author,                           // <-- 20
//! see VVWINFO::szRevisionNumber
RevisionNumber,
```

```
//! see VVWINFO::szProduced
Produced,
//! see VVWINFO::szAlbum
Album,
//! see VVWINFO::szArtist
Artist,
//! see VVWINFO::szComposer
Composer,
//! see VVWINFO::szCopyright
Copyright,
//! see VVWINFO::szCreationData
CreationData,
//! see VVWINFO::szDescription
Description,
//! see VVWINFO::szDirector
Director,
//! see VVWINFO::szDisclaimer
Disclaimer,                // <-- 30
//! see VVWINFO::szEncodedBy
EncodedBy,
//! see VVWINFO::szFullName
FullName,
//! see VVWINFO::szGenre
Genre,
//! see VVWINFO::szHostComputer
HostComputer,
//! see VVWINFO::szInformation
Information,
//! see VVWINFO::szMake
Make,
//! see VVWINFO::szModel
Model,
//! see VVWINFO::szOriginalArtist
OriginalArtist,
//! see VVWINFO::szOriginalFormat
OriginalFormat,
//! see VVWINFO::szPerformers
Performers,                // <-- 40
//! see VVWINFO::szProducer
Producer,
//! see VVWINFO::szProduct
Product,
//! see VVWINFO::szSoftware
Software,
//! see VVWINFO::szSpecialPlaybackRequirements
SpecialPlaybackRequirements,
//! see VVWINFO::szTrack
Track,
```

```
//! see VVWINFO::szWarning
Warning,
//! see VVWINFO::szURLLink
URLLink,
//! see VVWINFO::szEditData1
EditData1,
//! see VVWINFO::szEditData2
EditData2,
//! see VVWINFO::szEditData3
EditData3,                  // <-- 50
//! see VVWINFO::szEditData4
EditData4,
//! see VVWINFO::szEditData5
EditData5,
//! see VVWINFO::szEditData6
EditData6,
//! see VVWINFO::szEditData7
EditData7,
//! see VVWINFO::szEditData8
EditData8,
//! see VVWINFO::szEditData9
EditData9,
//! see VVWINFO::szVersionString
VersionString,
//! see VVWINFO::szManufacturer
Manufacturer,
//! see VVWINFO::szLanguage
Language,
//! see VVWINFO::szFormat
Format,                           // <-- 60
//! see VVWINFO::szInputDevice
InputDevice,
//! see VVWINFO::szDeviceModelNum
DeviceModelNum,
//! see VVWINFO::szDeviceSerialNum
DeviceSerialNum,
//! see VVWINFO::szReel
Reel,
//! see VVWINFO::szShot
Shot,
//! see VVWINFO::szTake
Take,
//! see VVWINFO::szSlateInfo
SlateInfo,
//! see VVWINFO::szFrameAttribute
FrameAttribute,
//! see VVWINFO::szEpisode
Episode,
```

```
//! see VVWINFO::szScene
Scene,                      // <-- 70
//! see VVWINFO::szDailyRoll
DailyRoll,
//! see VVWINFO::szCamRoll
CamRoll,
//! see VVWINFO::szSoundRoll
SoundRoll,
//! see VVWINFO::szLabRoll
LabRoll,
//! see VVWINFO::szKeyNumberPrefix
KeyNumberPrefix,
//! see VVWINFO::szInkNumberPrefix
InkNumberPrefix,
//! see VVWINFO::szPictureIcon
PictureIcon,
//! see VVWINFO::szProxyFile
ProxyFile,
//!
CustomMetadataBlockPointer,
//!
ImageInfo,                  // <-- 80
//! Unique ID
UMID,
//! Full XML of metadta
XML,
//! Custom inner XML for XMP - device dependant
XMP_Custom,
```

3.15.40.2 Numeric

```
NumericStart = 0x1000,
//! see VVWINFO::dwTimeCode
TimeCode,
//! see VVWINFO::dwUserBits
UserBits,
//! see VVWINFO::dwVITCTimeCode
VITCTimeCode,
//! see VVWINFO::dwVITCUserBits
VITCUserBits,       // <-- 4100
//! see VVWINFO::dwVITCLine3
VITCLine3,
//! see VVWINFO::dwPosterFrame
PosterFrame,
//! see VVWINFO::dwAFrame
AFrame,
//! see VVWINFO::dwAspectRatio
```

```
AspectRatio,
//! see VVWINFO::dwOriginalRate
OriginalRate,
//! see VVWINFO::dwOriginalScale
OriginalScale,
//! see VVWINFO::dwConversions
Conversions,
//! see VVWINFO::dwVersionNumber
VersionNumber,
//! see VVWINFO::dwFileSize
FileSize,
//! see VVWINFO::dwFileDate
FileDate,                   // <-- 4110
//! see VVWINFO::dwFileTime
FileTime,
//! see VVWINFO::dwSequenceNumber
SequenceNumber,
//! see VVWINFO::dwTotalStreams
TotalStreams,
//! see VVWINFO::dwTotalLength
TotalLength,
//! see VVWINFO::dwFilmManufacturerCode
FilmManufacturerCode,
//! see VVWINFO::dwFilmTypeCode
FilmTypeCode,
//! see VVWINFO::dwWhitePoint
WhitePoint,
//! see VVWINFO::dwBlackPoint
BlackPoint,
//! see VVWINFO::dwBlackGain
BlackGain,
//! see VVWINFO::dwBreakPoint
BreakPoint,                 // <-- 4120
//! see VVWINFO::dwGamma1000
Gamma1000,
//! see VVWINFO::dwTagNumber
TagNumber,
//! see VVWINFO::dwFlags
Flags,
//! see VVWINFO::dwTimeCodeType
TimeCodeType,
//! see VVWINFO::dwLTCTimeCodeType
LTCTimeCodeType,
//! see VVWINFO::dwVITCTimeCodeType
VITCTimeCodeType,
//! see VVWINFO::dwProdDate
ProdDate,
//End: v3.0
```

```
//! see VVWINFO::dwUniqueID
UniqueID,
//!
CustomMetadataBlockType,
CustomMetadataBlockSize,// <-- 4130
NorthSouthEastWest,
Latitude,
Longitude,
Exposure,
RedGain,
BlueGain,
WhiteBalance,
Matrix,
GreenGain,
```

3.15.40.3 *Read Only*

```
// Add elements here
//VVVID STRUCT
//! INTERNAL: Auto generated for XML output from #VVWVIDEO/#VVWAUDIO
VideoWidth = 0x10000,             // <-- 65536
//! XML tag name for width
VideoHeight,
//! XML tag name for height
VideoPlanes,
//! XML tag name for planes
VideoBitCount,
//! XML tag name for bit count
VideoCompression,          // <-- 65540
//! XML tag name for compression (fourcc)
VideoSizeImage,
//! XML tag name for size of the image in unsigned chars
VideoXPelsPerMeter,
//! XML tag name for X pels per meter
VideoYPelsPerMeter,
//! XML tag name for Y pels per meter
VideoClrUsed,
//! XML tag name for color elements used
VideoClrImportant,
//! XML tag name for
VideoReserved,
//! XML tag name for reserved array
VideoFccType,
//! XML tag name for four cc type (video/audio)
VideoFccHandler,
//! XML tag name for four cc handler
VideoFlags,
//! XML tag name for flags
```

```
VideoCaps,                          // <-- 65550
//! XML tag name for capabilities
VideoPriority,
//! XML tag name for priority
VideoLanguage,
//! XML tag name for language
VideoScale,
//! XML tag name for scale (fps = rate / scale)
VideoRate,
//! XML tag name for rate (fps = rate / scale)
VideoStart,
//! XML tag name for start frame
VideoLength,
//! XML tag name for the length in frames
VideoInitialFrames,
//! XML tag name for number of initial frames to load
VideoSuggestedBufferSize,
//! XML tag name for suggested maximum buffer size
VideoQuality,
//! XML tag name for quality
VideoSampleSize,                    // <-- 65560
//! XML tag name for recommended sample size
VideoEditCount,
//! XML tag name for number of edits done on this file
VideoFormatChangeCount,
//! XML tag name for number of format changes
VideoPitch,
//! XML tag name for video line pitch
VideoDrFlags,
//! XML tag name for internal drastic flags
VideoFileType,
//! XML tag name for drastic 'mft' file type
VideoResDrastic,
//! XML tag name for reserved drastic array of DWORDs
AudioType,
//! XML tag
AudioChannels,
//! XML tag
AudioFrequency,
//! XML tag
AudioBits,                          // <-- 65570
```

3.15.41 Get/SetClipMode

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?

SetValue&cmdalt=clipmode&position=0

Calls ValueXXX with gsClipMode. If equal to 1 then the channel is in Clip mode, if 0 the channel is in VTR mode.

3.15.42 Get/SetTCType

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?SetValue&cmdalt=tctype&position=2

Calls ValueXXX with gsTcType (Drop Frame, Non Drop Frame, PAL).

```
#define TC2_TCTYPE_MASK          0x000000FF
#define TC2_TCTYPE_FILM          0x00000001   // 24 fps
#define TC2_TCTYPE_NDF           0x00000002   // NTSC Non Drop Frame
#define TC2_TCTYPE_DF            0x00000004   // NTSC Drop Frame
#define TC2_TCTYPE_PAL           0x00000008   // PAL
#define TC2_TCTYPE_50            0x00000010   // PAL (double rate)
#define TC2_TCTYPE_5994          0x00000020   // NTSC 59.94fps 720p
#define TC2_TCTYPE_60            0x00000040   // NTSC 60fps 720p
#define TC2_TCTYPE_NTSCFILM      0x00000080   // NTSC FILM 23.97
```

3.15.43 Get/SetTCSource

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?GetValue&cmdalt=tcsource

Calls ValueXXX with gsTcSource (VITC, LTC, Control, Clip).

3.15.44 Get/SetAudioInput

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?SetValue&cmdalt=gsAudInSelect&position=lSetting&videochannels=0&audiochannels=lAudChannels&infochannels=0

ADD FUNCTION lAudIn

Get the current audio input.

//! Audio in/out unbalanced (RCA connector) high impedance at -10db
(cmdGetSetValue::gsAudInSelect cmdGetSetValue::gsAudOutSelect)
#define GS_AUDSELECT_UNBALANCED_10 0x001
//! Audio in/out unbalanced (RCA connector) high impedance at -4db
(cmdGetSetValue::gsAudInSelect cmdGetSetValue::gsAudOutSelect)
#define GS_AUDSELECT_UNBALANCED_4 0x002
//! Audio in/out balanced (XLR connector) 600ohm impedance at -10db
(cmdGetSetValue::gsAudInSelect cmdGetSetValue::gsAudOutSelect)
#define GS_AUDSELECT_BALANCED_10 0x010
//! Audio in/out balanced (XLR connector) 600ohm impedance at +4db
(cmdGetSetValue::gsAudInSelect cmdGetSetValue::gsAudOutSelect)
#define GS_AUDSELECT_BALANCED_4 0x020
//! Audio in/out digital single wire (cmdGetSetValue::gsAudInSelect cmdGetSetValue::gsAudOutSelect)
#define GS_AUDSELECT_SPDIF 0x100
//! Audio in/out digital balanced with clock (cmdGetSetValue::gsAudInSelect
cmdGetSetValue::gsAudOutSelect)
#define GS_AUDSELECT_AES_EBU 0x200
//! Audio in/out embedded in SDI or HD-SDI video signal (cmdGetSetValue::gsAudInSelect
cmdGetSetValue::gsAudOutSelect)
#define GS_AUDSELECT_EMBEDDED 0x400
//! No audio in/out available, or cannot be configured (cmdGetSetValue::gsAudInSelect
cmdGetSetValue::gsAudOutSelect)
#define GS_AUDSELECT_NONE

3.15.45 Get/SetAudioInputLevel

**http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?
SetValue&cmdalt=gsAudInputLevel&position=lSetting&videochannels=0&audiochannels=lAud
Channels&infochannels=0**

Get the current audio input level. This requires capture hardware that supports input level setting.

3.15.46 Get/SetAudioOutput

**http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?**

SetValue&cmdalt=gsAudOutSelect&position=lSetting&videochannels=0&audiochannels=lAudChannels&infochannels=0

Get the current audio Output – See Get/SetAudioInput

3.15.47 Get/SetAudioOutputLevel

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?SetValue&cmdalt=gsAudOutputLevel&position=lSetting&videochannels=0&audiochannels=lAudChannels&infochannels=0

Get the current audio output level.

3.15.48 Get/AudioPeakRMS

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?GetValue&cmdalt=gsAudWavePeakRMS

Returns the RMS and Peak audio levels of the input (stop/record) or output (play/pause).

3.15.49 Get/SetVideoInput

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?SetValue&cmdalt=gsVidInSelect&position=lSetting

Get the current video input.

//! Standard NTSC or PAL composite video (cmdGetSetValue::gsVidInSelect cmdGetSetValue::gsVidOutSelect)
#define GS_VIDSELECT_COMPOSITE 0x001
//! SVHS/S-Video four wire NTSC or PAL video (cmdGetSetValue::gsVidInSelect cmdGetSetValue::gsVidOutSelect)
#define GS_VIDSELECT_SVIDEO 0x002
//! Secondary NTSC or PAL video (often monitor selection) (cmdGetSetValue::gsVidInSelect cmdGetSetValue::gsVidOutSelect)
#define GS_VIDSELECT_COMPOSITE_2 0x004

```
//! BetaCam level YCrCb NTSC or PAL video (cmdGetSetValue::gsVidInSelect
cmdGetSetValue::gsVidOutSelect)
#define GS_VIDSELECT_COMPONENT_YUV                    0x010
//! Panasonic M2 level YCrCb NTSC or PAL video (cmdGetSetValue::gsVidInSelect
cmdGetSetValue::gsVidOutSelect)
#define GS_VIDSELECT_COMPONENT_YUV_M2                    0x020
//! SMPTE standard level YCrCb NTSC or PAL video (cmdGetSetValue::gsVidInSelect
cmdGetSetValue::gsVidOutSelect)
#define GS_VIDSELECT_COMPONENT_YUV_SMPTE          0x040
//! RGB at video standard rate (cmdGetSetValue::gsVidInSelect cmdGetSetValue::gsVidOutSelect)
#define GS_VIDSELECT_COMPONENT_RGB                    0x080
//! D1 Serial Digital or HDSDI video (cmdGetSetValue::gsVidInSelect
cmdGetSetValue::gsVidOutSelect)
#define GS_VIDSELECT_D1_SERIAL                        0x100
//! D1 Serial Parallel video (cmdGetSetValue::gsVidInSelect cmdGetSetValue::gsVidOutSelect)
#define GS_VIDSELECT_D1_PARALLEL                      0x200
//! SDTI/SDI including high speed transfer video (cmdGetSetValue::gsVidInSelect
cmdGetSetValue::gsVidOutSelect)
#define GS_VIDSELECT_SDTI                             0x400
//! No video available or no configurable settings (cmdGetSetValue::gsVidInSelect
cmdGetSetValue::gsVidOutSelect)
#define GS_VIDSELECT_NONE                             0
```

3.15.50 Get/SetVideoInputSetup

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?SetValue&cmdalt=gsVidInSetup&position=lSetting

Get the current video input's 'Setup' TBC setting. This requires capture hardware that has a built in time base corrector.

3.15.51 Get/SetVideoInputVideo

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?SetValue&cmdalt=gsVidInVideo&position=lSetting

Get the current video input's 'Video' TBC setting. This requires capture hardware that has a built in time base corrector.

3.15.52 Get/SetVideoInputHue

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?SetValue&cmdalt=gsVidInVideo&position=lSetting

Get the current video input's 'Hue' TBC setting. This requires capture hardware that has a built in time base corrector.

3.15.53 Get/SetVideoInputChroma

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?SetValue&cmdalt=gsVidInChroma&position=lSetting

Get the current video input's 'Chroma' TBC setting. This requires capture hardware that has a built in time base corrector.

3.15.54 Get/SetVideoTBCSetup

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?SetValue&cmdalt=gsVidSetup&position=lSetting

Get the current global TBC's 'Setup' setting. This requires capture hardware that has a built in time base corrector.

3.15.55 Get/SetVideoTBCVideo

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?SetValue&cmdalt=gsVidVideo&position=lSetting

Get the current global TBC's 'Video' setting. This requires capture hardware that has a built in time base corrector.

3.15.56 Get/SetVideoTBCHue

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?SetValue&cmdalt=gsVidHue&position=lSetting

Get the current global TBC's 'Hue' setting. This requires capture hardware that has a built in time base corrector.

3.15.57 Get/SetVideoTBCChroma

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&channel=0&XMLMediaCmd?SetValue&cmdalt=gsVidChroma&position=lSetting

Get the current global TBC's 'Chroma' setting. This requires capture hardware that has a built in time base corrector.

3.16 Full MediaCmd AJAX/XML Access

This access method allows our javascript or php application to access all the same functions used by Drastic's GUIs from an html interface. The basic form of the commands is:

http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd&<mediacmd>

The <mediacmd> is a series of ampersand delimited (&) commands and modifiers. Normally this command will be sent via an HTTPObject, and will return synchronously or asynchronously a standard XML return that can be parsed. To send a command in AJAX/Javascript, you will first need to instantiate an HTTPObject to send it through. Here is an HTTPObject instantiation that will work in most browsers:

```
// Create an HttpObject
function getHTTPObject()
{
      var xmlhttp;

      /*@cc_on
            @if (@_jscript_version >= 5)
                  try
```

```
                {
                      xmlhttp = new ActiveXObject("Msxml2.XMLHTTP");
                }catch (e)
                {
                      try
                      {
                            xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
                      } catch (E)
                      {
                            xmlhttp = false;
                      }
                }
          @else
                xmlhttp = false;
          @end @*/

     if (!xmlhttp && typeof XMLHttpRequest != 'undefined')
     {
          try
          {
                xmlhttp = new XMLHttpRequest();
          } catch (e)
          {
                xmlhttp = false;
          }
     }
     return xmlhttp;
}

// Instantiate the various HTTP Objects
var _xmlHttp = getHTTPObject(); // Create the HTTP Object
```

Once the HTTPObject is instantiated into a variable, the variable (_xmlHttp in this case) can be used to call the DDR and send and receive MediaCmds. These commands can be sent synchronously (the command will complete and return the XML immediately) or asynchronously (the command will process, but return immediately. Later a callback will be called with the return XML data). Either way the return will be the same.

To send a command synchronously (return after processing) without using the return:

```
function play()
{
      // Build the URL to connect to
      var url = "netx?request=set&client=0&group=sdi&sdicmd&Play”;
      // Open a connection to the server
      _xmlHttp.open("GET", url, false);   // indicates sync call
      // Send the request
      _xmlHttp.send(null);
}
```

For a command that is sent synchronously, but the return needs to be processed, the call is very similar:

```
function getClipMode()
{
      // Build the URL to connect to
      var url = "netx?
request=set&client=0&group=sdi&sdicmd&GetValue&position=0&cmdalt=ClipMode&Flags=-
1”;
      // Open a connection to the server
      _xmlHttp.open("GET", url, false);   // indicates sync call
      // Send the request
      _xmlHttp.send(null);
      // Get the clip mode out of the XML response
      var xmlobject = _xmlHttpMode.responseXML;
      if(xmlobject == null)   {
            return;
      }
      // Get MediaCmd return (in XML)
      var mCmd = xmlobject.getElementsByTagName("MediaCmd");
      if(mCmd[0])
      {
            // Return the current mode
            return mCmd[0].getElementsByTagName("Position")
[0].getAttribute("Value")
        }
      return “errorValue”;
}
```

A typical XML return would look like this:

<insert mediacmd XML return here>

Often, to maximize user responsiveness, or to allow for long commands to process, commands need to be sent asynchronously. The asynchronous version of the command is essentially the same as the synchronous with processing version, except the send and return are divided into separate functions:

```
function getClipMode()
{
      // Build the URL to connect to
      var url = "netx?
request=set&client=0&group=sdi&sdicmd&GetValue&position=0&cmdalt=ClipMode&Flags=-
1”;
      // Open a connection to the server
      _xmlHttp.open("GET", url, false);   // indicates sync call
      // Setup a function for the server to run when it's done
      _xmlHttp.onreadystatechange = updateClipMode;
      // Send the request
      _xmlHttp.send(null);
}
// A response to an 'Mode' request has been received
function updateMode()
      if (_xmlHttpMode.readyState == 4)
      {
            // A complete response has been received
            // Get the clip mode out of the XML response
            var xmlobject = _xmlHttpMode.responseXML;
            if(xmlobject == null)   {
                  return;
            }
            // Get MediaCmd return (in XML)
            var mCmd = xmlobject.getElementsByTagName("MediaCmd");
            if(mCmd[0])
            {
                  // Return the current mode
                  return mCmd[0].getElementsByTagName("Position")
[0].getAttribute("Value")
                  }
      }
      return “errorValue”;
}
```

3.16.1 sdicmd main commands

The first parameter of the http://localhost:1080/netx?request=set&client=0&group=sdi&sdicmd& (following the question mark) must be one of the following commands:

Stop – Full stop/all stop/E to E
Pause – Pause on current frame, seek or load
Play – Play, either at normal speed or shuttle speeds. May also load and seek.
Record – Record to the disk or tape
RecStop – Prepare for a record
Eject – Eject the current tape or media
Transfer – Transfer to/from an internal channel and an external channel
Insert – Insert media into the clip bin or time code space
Blank – Remove media from the clip bin or time code space
Delete – Delete media from the storage and blank it
Trim – Alter a clip or time code space edit
ChanSelect – Change the currently selected channels
GetState – Get the current channel state
SetState – Set the current channel state
GetValue – Get a setup value
ValueSupported – See if a setup value is supported
SetValue – Change a setup value
Error – Report an error
Terminate – Kill the current operation
Abort – Abort the current operation

3.16.2 Basic Command (sdicmd)

netx?request=set&client=0&group=sdi&sdicmd&<modifiers>

With these commands a number of modifiers are available. Each modifier must be separated by an ampersand (&) on the command line.

channel=%d – specify the channel this command should be sent to
position=%s – set the position element for a command
1:00:00:00 – go to one hour
+5 – go forward from the current location 5 frames
-5:00 – go backward from the current location 5 seconds
1800 – go to one minute (specified as 1800 frames, non drop frame time code)
start=%s – set the start element (see position for format)
end=%s – set the end element (see position for format)
speed=%d – set the speed element for a command
65520 – normal forward play (100%)
-65520 – reverse play
32760 – half play speed (50%)
-655200 – 10 times reverse speed
0 - pause (no play)
timems – millisecond time for the command
cmdalt – set the cmdalt element of the mediacmd
videochannels – which video channels to use (bitwise)
audiochannels – which audio channels to use (bitwise)
infochannels – which information channels to use (bitwise)
clipid – 8 character clip identifier
filename – filename for the command
string – string to be used in the command

There are a number of flags that may be used, just like the elements above

Deferred – wait for previous command to complete before new this command
OverrideDeferred – override a previous deferred command
Loop – Loop whole clip, or a start/end subset
AllIDs – Command should affect all available clip ids
NoClipFiles – Ignore clip space clips
NoTCSpaces – Ignore conform space files
IsShuttle – The command should be interpreted as a shuttle, even for normal play
UsingCurrent – Use the current start/end/position
UseFrameCount – Use the absolute frame count, not the time code values

Fields – Use fields, if not a progressive signal format
Ripple – When removing a file, ripple the following files back
Trigger – Wait for a trigger
Preview – Doing a preview, not a full play
Test – Don't do the command, just see if it exists
NoReturn – Don't return any information from the command

3.16.3 sdicmd Examples

netx?request=set&client=0&group=sdi&sdicmd&play
– Normal play

netx?request=set&client=0&group=sdi&sdicmd&play&speed=32760
– Play at 50% forward speed

netx?request=set&client=0&group=sdi&sdicmd&play&speed=-65520
– Play at 100% reverse play speed

netx?request=set&client=0&group=sdi&sdicmd&play&start=1:00&end=4:00&loop
– Play from one second to four seconds in a loop

netx?request=set&client=0&group=sdi&sdicmd&pause
– Pause the channel

netx?request=set&client=0&group=sdi&sdicmd&stop
– Stop (E to E passthrough) the channel

netx?request=set&client=0&group=sdi&sdicmd&pause&position=1:00:00
– Seek to one minute

netx?request=set&client=0&group=sdi&sdicmd&record&clipid=newrec&end=5:00
– Record a new file name 'newrec' which will be five seconds long

3.16.4 Dealing with Picon Images

Server Mode, clip: Kroatien, file: KroatienMovie.mov

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&SetValue&cmdalt=1000000&clipid=Kroatien&position=200
– Make a new picon from frame 200 of the clip Kroatien
– result name: KroatienMovie.picon.jpg

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&GetValue&cmdalt=1000000&clipid=Kroatien&position=ffffffff
– Return the actual file name of the picon file (char elem 9)
– result name: Kroatien.picon.jpg

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&GetValue&cmdalt=1000000&clipid=Kroatien&position=4294967295
– Return the size of the picon file in the Position elements
– result: dwPosition = 7900

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&GetValue&cmdalt=1000000&clipid=Kroatien&position=1
– Return the actual bytes of data for the JPEG picon frame in arbID
– result: Not available in HTTP, have to use C/C++

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&SetValue&cmdalt=1000000&filename=V:\Media\KroatienMovie.mov&position=100
– Make a new picon from frame 100 without associating it with the clip
– result name: KroatienMovie.picon.jpg
(not normally used, conflicts with VTR tape mode picon)

VTR Tape Mode, Time line 00:00:01:00 Kroatien.mov?

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&SetValue&cmdalt=1000000&filename=V:\Media\Kroatien.mov&position=1000
– Make a new picon from the frame at position 1000, default for file
– result name: Kroatien.picon.jpg

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&GetValue&cmdalt=1000000&filename=V:\Media\Kroatien.mov&position=ffffffff
– Return the actual file name of the picon file (char elem 9)
– result name: Kroatien.picon.jpg

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&GetValue&cmdalt=1000000&filename=V:\Media\Kroatien.mov&position=4294967295
– Return the size of the picon file in the Position elements
– result: dwPosition = 7900

http://localhost:1080/netx?
request=set&client=0&group=sdi&sdicmd&GetValue&cmdalt=1000000&filename=V:\Media\Kroatien.mov&position=1
– Return the actual bytes of data for the JPEG picon frame in arbID
– result: Not available in HTTP, have to use C/C++

3.17 Special XML Access Commands

XML Returns. These are to be used with AJAX/DOM pages.

http://localhost:1080/netx?request=set&client=0&group=sdi&sdigetstatus
– Returns an XML package including state, speed, position start and end points.

http://localhost:1080/netx?request=set&client=0&group=sdi&sdinextclip
– Returns an XML package with all the clip information. Used to retrieve the clip bin information

http://localhost:1080/netx?request=set&client=0&group=sdi&sdiclipinfo
– Returns an XML package with all the clip information. Used to retrieve information on a specific clip

http://localhost:1080/netx?request=set&client=0&group=sdi&sdiedlstate
– Used in conjunction with sdiedlinfo to retrieve the time code space edits. The command will always be sdiedlstate?position=#&videochannels=#&audiochannels=#&infochannels=#.

http://localhost:1080/netx?request=set&client=0&group=sdi&sdiedlinfo
– Used in conjunction with sdiedlstate to retrieve the time code space edits.

Here is a basic EDL retrieval session:

Call··· ·Position··· ·Start··· ·End ··· ·V ··· ·A ··· ·I ··· ·File Name··· ·Comment
sdiedlinfo··· ·0···
0··· ·0··· ·0···
Restart list at 0
return info··· ·0··· ·0··· ·300··· ·1··· ·2··· ·0··· ·file1.mov··· ·10 sec VA2 from file1
sdiedlstate··· ·0···
0··· ·0··· ·0···
First state sent in above
return state··· ·0···
1··· ·2··· ·0···
Used clip channels to pass back into Info
sdiedlinfo··· ·0···
1··· ·2··· ·0···
Copy of the return of VVWXMLEDLState above
return info··· ·0··· ·0··· ·150··· ·0··· ·1··· ·0··· ·file2.wav··· ·5 sec A1 from file2
sdiedlstate··· ·0···
1··· ·2··· ·0···
Use the return of the last VVWXMLEDLState

return state··· ·0···
1··· ·3··· ·0···
These are the channels used so far
sdiedlinfo··· ·0···
1··· ·3··· ·0···
Copy of the return of VVWXMLEDLState above
return info··· ·150··· ·150··· ·210··· ·0··· ·1··· ·0··· ·file3.wav··· ·2 sec A1 from file3
sdiedlstate··· ·0···
1··· ·3··· ·0···
Use the return of the last sdiedlstate
return state··· ·150···
0··· ·1··· ·0···
All edits completed before 150

Take the MEDIACMD struct returned from sdiedlstate and find the next active clip. For the first clip in the timeline, send all zeroes. Other than the first call, all calls should include the position/channel bits from the previous sdiedlstate call and (other than first call) sdiedlstate should be called immediately before sdiedlinfo.

http://localhost:1080/netx?request=set&client=0&group=sdi&sdinextdirentry
– Used to retrieve the directory structure.
Takes 2 parameters:
The base directory you are getting the listing for
The last directory entry returned

Assuming you had a directory structure that looked like this:

\Record\
\Record\Test.wav
\Record\Test.avi
\OfflineMedia\
\OfflineMedia\EmptyDir\
\OfflineMedia\retry.doc
\OfflineMedia\big.tga
\LocalMedia\AnotherDir\
\LocalMedia\test.aiff

The first call would only include the parameter '\'
http://localhost:1080/netx?request=set&client=0&group=sdi&sdinextdirentry&\
Returns: <locator>\Record</locator>

This will return the first FileDir XML structure that will include the first locator. To get the next item, return the same base path plus the new locator.

http://localhost:1080/netx?request=set&client=0&group=sdi&sdinextdirentry&\&\Record
Returns: <locator>\OfflineMedia</locator>
http://localhost:1080/netx?request=set&client=0&group=sdi&sdinextdirentry&\&\OfflineMedia
Returns: <locator>\LocalMedia</locator>
http://localhost:1080/netx?request=set&client=0&group=sdi&sdinextdirentry&\&\LocalMedia
Returns: <locator>END OF LIST</locator>

To descend into a sub directory, use the sub directory as the base path. To see what is in \Record

http://localhost:1080/netx?request=set&client=0&group=sdi&sdinextdirentry&\Record\
Returns: <locator>\Record\..</locator>
http://localhost:1080/netx?request=set&client=0&group=sdi&sdinextdirentry&\Record\&\Record\..
Returns: <locator>\Record\Test.wav</locator>
http://localhost:1080/netx?request=set&client=0&group=sdi&sdinextdirentry&\Record\&\Record\
Test.wav
Returns: <locator>\Record\Test.avi</locator>
http://localhost:1080/netx?request=set&client=0&group=sdi&sdinextdirentry&\Record\&\Record\
Test.avi
Returns: <locator>END OF LIST</locator>

http://localhost:1080/netx?request=set&client=0&group=sdi&sdifileinfo
– Used to retrieve information on a specific file.

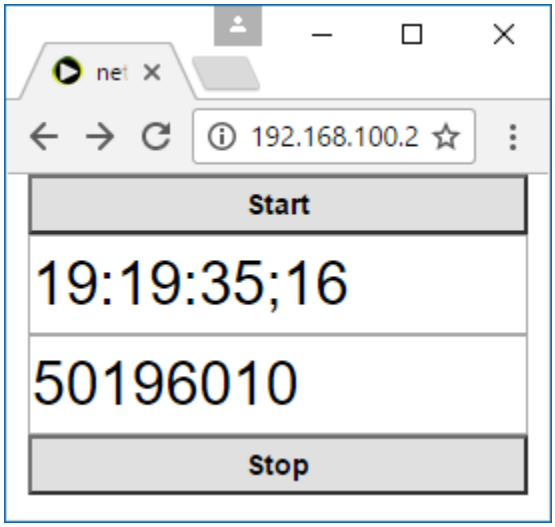
http://localhost:1080/netx?request=set&client=0&group=sdi&sdigeterrormsg&#
– Used to return one error message from the current list. The first call will not include an error number (just sdigeterrormsg). This will return an ErrorNumber to use to get the next message (sdigeterrormsg&202 for instance), as will each subsequent call. When all the error messages have been returned, it will return an ErrorNumber of -1.

4 Net-X-Time-Code

The Net-X-Time-Code server is a server that captures a UDP or RTP stream of time code metadata and converts it to an AJAX compatible HTTP server. The server records the latest time code and userbits, which can then be viewed within the default HTML page, or requested via AJAX requests. The base command for Net-X-Time-Code is “/netxtc?”. The base address is your machine's IP at port 1080. E.g.:

http://127.0.0.1:1080

Which should bring up the default page and display a running time code.



4.1.1 Set A Variable

Set one of the get/set variables described below

http://127.0.0.1:1080/netxtc?request=set&address=238.255.99.99

4.1.2 Get A Variable

Get one of the get/set variables described below

http://127.0.0.1:1080/netxtc?request=set&address=

4.1.3 Get/Set Variables

All – used to return all the variables
name – file name, not used
port – incoming stream port
protocol – RTP or UDP
type – file type, not used
state – current state
desc – stream description
tc – time code
ub - userbits
frames – time code in frames

ms - milliseconds

4.1.4 Start Capturing Time Code

Start capturing time code from the time code stream source

http://127.0.0.1:1080/netxtc?request=command&start=

4.1.5 Stop Capturing Time Code

Stop capturing time code from the time code stream source

http://127.0.0.1:1080/netxtc?request=command&stop=

4.2 Net-X-Copy Command Line

netxcopy: Setup DrasticDDR for specific configurations (v7.1.0.230)
netxcopy -s <sourcefile> -t <targetfile> [-a <ackfile>] -p <profile> [-in <00:01:00:00> -out <00:02:00:00> -fg]

-s <sourcefile> - The source file name and path
-t <targetfile> - The target file name and path
-a <ackfile> - The ACK file name and path. This is the XML acknowledgment file made after a copy
-p metadata - Return XMP metadata for a media file
-p cardinfo - Return XML/JSON info on clips on a camera card
-p dir - Return XML/JSON directory listing
-p clipfiles - Return all the files associated with a media clip
-p md5 - Calculate or check the MD5 value source, send compare string as target
-p <profile> - Profile to use. Current profiles include:
> **copy** - copy the whole file
> **wrap** - re wrap file or part of a file
> **index** - create an RTIndex for a file
> **getCopyInOut** - get the extents required for a pfr, or use them with a temp file
> **mp3-128kbps** - Audio MP3 file
> **wave** - Audio wave file
> **caption-#** - Extract captions from a file (see types below)
> **mov-YCbCr8Bit** - QuickTime MOV 8 bit uncompressed YCbCr file
> **mov-dvcprohd** - QuickTime MOV DVCPro HD (1080/720)

> **mp4-h264** - MPEG-4 h264 AAC Audio
> **mxf-xdcam-720p** - True XDCam MXF 8 channel audio
> **mxf-dvcprohd-720p** - MXF DVCPro HD 720p
> **mxf-xdcam-1080i** - True XDCam MXF 1080i 8 channel audio
> **mxf-dvcprohd-1080i** - MXF DVCPro HD 1080i 29/25 fps
> **mxf-OP1a-MPEG** - OpenMXF XDCam MPEG-2 16 channel audio
> **mxf-OP1a-h264** - MXF h.264
> **mxf-OP1a-HDF** - MXF MPEG-2 HDF Standard
> **mxf-as-11-sd-pal-dpp** - MXF AS-11 SD PAL DPP
> **mxf-as-11-sd-ntsc-dpp** - MXF AS-11 SD NTSC DPP
> **mxf-as-11-hd-dpp** - MXF DPP AS-11 AVCi HD
> **mxf-OP1a-JPEG2K** - Samma style JPEG2000 YCbCr
> **mxf-AS-02-h264-10** - 10 bit 50 Mbs h.264 in AS-02 MXF
> **OP1a_HBR_50** - OP1a MXF h264 4:2:2 10 bit
> **mov-proreshq** - QuickTime MOV ProRes HQ
> **mov-proreslt** - QuickTime MOV ProRes LT
> **mov-prores422** - QuickTime MOV ProRes 422
> **mov-prores444** - QuickTime MOV ProRes 444(4)
> **scaledown2000k** - MP4 264 960x540, 2mbs, AAC
> **scaledown500k** - MP4 264 480x272, 0.5mbs, AAC
> **hd1080-5000kbs** - MP4 HD 1080 with a target bitrate of 5 mbs
> **hd720-2500kbs** - MP4 HD 720p with a target bitrate of 2.5 mbs
> **hd360-1250kbs** - MP4 HD 360p with a target bitrate of 1.25 mbs
> **h264-7500kbs** - MP4 Any resolution with a target bitrate of 7.5 mbs
> **Proxy-h264-5000kbs** - MP4 high quality proxy for web
> **LBR-h264-10000kbs** - Low bit rate, high quality local MP4
> **HEVC-h265-10000kbs** - Low bit rate, high quality HEVC MP4
> **DASH-MP4-Multibitrate** - Multi bitrate MP4s with DASH files
> **HLS-TS-Multibitrate** - Multi bitrate TS streams with M3U8 files
> **MP4-MultiOutput** - Multi MP4 with optional burn in files
> **TS-TR-01-JPEG-2000** - TR-01 JPEG-2000 transport stream
> **TS-MPEG2** - MPEG-2 4:2:0/passthrough transport stream
> **TS-h264** - h.264 4:2:0/passthrough transport stream
> **mp4-XAVC-S_4_2_0** - MP4 Sony XAVC-S 4:2:0
> **mp4-XAVC-S_4_2_2** - MP4 Sony XAVC-S 4:2:2
> **aces** - ACES image files
> **MXF-RDD-25** - MXF RDD-25 Proxy
> **dnxhd-mxf-720p** - DNxHD 720p 50, 59, 60
> **dnxhd-mxf-1080p** - DNxHD 1080p 25, 29
> **dnxhd-mxf-1080i** - DNxHD 1080i 25, 29
> **dnxhr-mxf-10-hq** - DNxHR High Quality 10 bit
> **dnxhr-mxf-8-hq** - DNxHR High Quality 8 bit
> **dnxhr-mxf-sq** - DNxHR Standard Quality

> **dnxhr-mxf-lq** - DNxHR Low Quality
> **amt3-HQX_10** - AMT 3 DNx HQX 10 Bit
> **amt3-HiQuality** - AMT 3 DNx High Quality
> **amt3-StandardQuality** - AMT 3 DNx Standard Quality
> **amt3-LowQuality** - AMT 3 DNx Low Quality
> **amt3-DNxHD36** - AMT 3 1080 DNxHD 36
> **amt3-Consolidate** - AMT 3 Any Avid Supported Codec

-type mxf-op1a -- the exact file type to write, otherwise auto
> **mxf-op1a** - standard OP1a
> **mxf-sonyhd** - Sony XDCam compatible
> **mxf-as02** - AS - 02 spec MXF
> **mxf-avid** - Avid OP-Atom (Drastic->bin)
> **mxf-amt** = Avid OP-Atom (Avid->aaf)
> **mp4-fmp4** - Fragmented MP4(normal MP4 if not set)
> **mp4** - MPEG-4
> **mov** - QuickTime MOV
> **ts** - Transport stream

-in <00:01:00:00> - the starting point for the output file in time code or absolute position
-out <00:02:00:00> - the ending point for the output file in time code or absolute position
-absin <200> - the absolute (zero based) start time for the output file (overrides -in)
-tcoffset <00:01:00:00> - Offset the timecode by this amount
-tc <01:00:00:00> - Replace the output timecode starting with this timecode
-ub <ABCD0123> - Replace the output userbits with these userbits
-absout <400> - the absolute (zero based) end time, exclusive, for the output file (overrides -out)
-width <width> - output width of the video (only for arbitrary codecs like h264, hevc and prores)
-height <height> - output height of the video (if not set, the input size or codec size will be used)
-copy - make a copy of the file section we need, instead of reading directly
-dest - folder or folder and file name for the temp file when using copy
-dest <folder> - folder or folder and file name for the temp file when using copy
-cc <mcc/scc file> - replacement closed caption file>
-afile <path-audio-file> - replacement source audio track<s>
-v <path-video-file> - replacement source video track
-stereo - force a stereo pair (mixdown) output
-aroute <12345678/auto> - route channels to specific outputs
-uuid <uuid string> - override the UUID of the file with this one
-kilobitrate <kbit rate> - override the kilo bit rate
-h26xprofile <baseline / main / high / high10 / high422 / high444> - override the profile type
-h26xlevel <51> - override the level
-encodemode <0 / 1> - 0 normal, 1 fastest
-gopsize <15> - size of encoded gop
-tempfolder - Temporary folder to store partial file
-alignment - Alignment value for any temporary partial files, for GetCopyInOut profile
-flags flag - Extra flags for special operations

allSubAudioTracks,aiProfileID,audiofirstonly,ccExtractChannel,
ccOutputDirectory, convert708to608,eMamAssetID,eMamCategory,eMamProxyID=,
extractCCLine21,fixOut,forceCCVidStream,forceOutputFramerate,fpsCompensation,
frameAccurate,interplay,interpretFileEncoding,markFirstTime,multi,notifyeMam,offseTC,
overrideTC,proxyAfterRecord,proxyDisable,proxyEnable,proxyRecord,
sampleaccurate,shuffleAudio,testMode,timeConversion,zeroBasedOutput

-m - Save the metadata in an XMP file

-fg - force the GUI on

-fc - force command line

NOTE: the parameters in [square brackets] are optional.

For JPEG picons/thumbnails

-pisrc <file-to-picon> - source for the image

-pidst <where-to-make-picon> - target folder and name

-pisize <size> - size of picon, 100%

-piframe <frame-offset-into-file> - frame to use to make the picon

-piskip <number-of-frames-to-skip> - if set, make a picon of each frame at this distance for the whole file

-width <width> - output width of the picon image

For Caption Extraction, these are the numbers to use with -p caption-#

> caption-# - eg caption-0=mcc, caption-6=SCCAUTO, etc

0=MCC,
1=AAF,
2=TEXT,
3=SCC,
4=SCCDF,
5=N0,
6=SCCAUTO,
100=Avid Caption,
101=Belle Nuit Subtitler,
102=CapMaker Plus,
103=Cheetah Caption,
104=Csv,
105=Csv2,
106=Csv3,
107=D-Cine SMPTE 2007,
108=D-Cine SMPTE 2010,
109=EBU STL,
110=Final Cut Pro Xml,
111=Final Cut Xml Gap,
112=Final Cut Pro X Xml,

113=Final Cut Pro XCM,
114=Flash Xml,
115=iTunes Timed Text,
116=JSON,
117=JSON Type 2,
118=JSON Type 3,
119=JSON Type 4,
120=QuickTime text,
121=SAMI,
122=SAMI modern,
123=Scenarist,
124=DVD_SCC,
125=DVD_SCCDF,
126=SoftNi sub,
127=Sony DVDArchitect,
128=Sony DVD Exp dur,
129=Sony DVD line/dur,
130=Sony DVD Tabs,
131=SubRip,
132=Sub Station Alpha,
133=Swift V2,
134=Timed Text 1.0,
135=TT 2006-04 .dfxp,
136=TT 2006-04 .ttml,
137=TT 2006-04 .xml,
138=WebVTT,
139=YT Annotations,
140=YouTube sbv,
141=YouTube Transcript,
142=D-Cinema interop,
143=WebVTT-webvtt,
144=SMPTE-TT 2052,
145=Netflix Timed Text,

-q	Quiet mode.
-l	Loud mode.
-r 01:00:00:00	Override timecode start
-R -00:30:00:00	Offset timecodes (e.g. subtract half an hour)
-0	Zero based timecode output
-1	Mark this first timecode in file
-h #	eMAM source clip ID
-s	Force CC in video stream
-X	Extract CC from line 21 (SD only)

-M # CC channel to extract (def: 1)
-C Use 608 compatibility bytes (not 708)
-c # Convert closed captioning to
-f # Force output frame rate (23, 24, 25, 29, 30)
-g # Compensate conversion (2324, 2423, 2930, 3029)
-o <out-dir> CC output directory
-a # Interpret input file using encoding
0=Auto, 1=UTF-8, 2=UTF-16, 3=IBM-850, 4=IBM-860, 5=Windows-1252

h26x Profiles:
10, 11, 12, 13, 20, 21, 22, 30, 31, 32, 40, 41, 42, 50, 51, 52

h26x Levels:
66 – Baseline
77 – Main
88 – Extended
100 – High
110 – High 10
122 – High 4:2:2
144 – High 4:4:4
166 – Advanced 4:4:4 Intra
188 – Advanced 4:4:4

NOTE: the parameters in [square brackets] are optional.
For further examples see:
https://www.drastic.tv/support-59/supporttipstechnical/295-netxcopy-pfr-clip-metadata
https://www.drastic.tv/support-59/supporttipstechnical/296-netxcopy-pfr-command-line
https://www.drastic.tv/support-59/supporttipstechnical/298-netxcopy-metadata-extraction
https://www.drastic.tv/support-59/supporttipstechnical/299-netxcopy-metadata-cardinfo

4.3 Discontinuity Sources In Net-X-Code

Net-X-Code tracks and saves any issues it encounters during capture as discontinuities. These are saved in the ACK output file with their time, time code, absolute location and source. The possible sources are:

- PID continuity counter discontinuity
  - video (DISCON_VIDEO)
  - audio

    - dolby
    - timecode
    - metadata
    - metadata cc
    - metadata tc
    - video pts
    - audio pts
- Time code dup/missing discontinuity
- UUID in metadata change
- No packets for over 100 millisecond discontinuity

Most of these are 'true' discontinuities, where there is an error in the transport layer that denotes a problem upstream. The one exception is the time code discontinuity. Because the LBR/ABR are muxed, missing video/audio frames may not show up as discontinuities, as they occurred pre mux. The only indication (other than broken video/audio decodes) is that the time code jumps, so it is important to treat this as a full discontinuity.

4.3.1 Discontinuity Reporting

Version 5 and 6/7 have slightly different reporting of discontinuities for historical reasons. Here are the types and how they appear:

DISCON_VIDEO:

5: `"video" "Video discontinuity pid=%d %d!=%d"`

6: `"video" "Video packet-b discontinuity pid=%d last=%d current=%d ms=%d", "Video discontinuity at wrap %d frames - No Audio found %d Video Q(%d) %llu-%llu Audio Q(%d) %llu-%llu", "Video discontinuity - No Audio found %d Video Q(%d) %llu-%llu Audio Q(%d) %llu-%llu"`

DISCON_AUDIO:

5: `"audio" "Audio discontinuity pid=%d %d!=%d"`

6: `"audio" "Audio packet-b discontinuity pid=%d last=%d current=%d. Packet will be repeated. ms=%d", "Audio discontinuity - no audio length = %llu, %llu, %llu, %llu, %d, %d Audio Handler %08X", "Audio discontinuity - no audio for video ((audio chan %d pts %llu + %d length (%llu %.2f) = %llu) V{%llu - %llu} A{%llu - %llu} (%llu - %llu = %llu)", "Audio %d discontinuity at wrap %d frames - No Video found Video Q(%d) %llu-%llu Audio Q(%d) %llu-%llu"`

DISCON_DOLBY:

5: `"audio" "Dolby discontinuity pid=%d %d!=%d"`

6: `"audio" "Dolby packet-b discontinuity pid=%d last=%d current=%d ms=%d"`

DISCON_META_TC:

5: `"meta tc" "TCMeta discontinuity pid=%d %d!=%d"`

6: `"meta tc" "TCMeta packet-b discontinuity pid=%d last=%d current=%d ms=%d"`

DISCON_METADATA:

5: `"metadata"  "Metadata discontinuity pid=%d %d!=%d", "UMID discontinuity new UMID=<%S>"`

6: `"metadata"  "Metadata packet-b discontinuity pid=%d last=%d current=%d ms=%d", "UMID discontinuity new UMID=<%S>"`

DISCON_META_CC:

5: `"meta cc"  "CCMeta discontinuity pid=%d %d!=%d"`

6: `"meta cc"  "CCMeta packet-b discontinuity pid=%d last=%d current=%d ms=%d"`

DISCON_TIMECODE:

5: `"timecode"  "CurrentTimeCode: %lu"`

6: `"timecode"  "CurrentTimeCode: %lu"`

DISCON_PTS_VIDEO:

5: `"video"  "PTS discontinuity - drop audio before video = %llu, %llu, %llu, %llu, %d, %d\n"`

6: `"video"  "Video B Frame discontinuity - drop  video = %llu, %llu, %llu, %llu, %d", "Video Group discontinuity - drop  video = %llu, %llu, %llu, %llu, %d", "PTS discontinuity - audio gap drop video = %llu, %llu, %d"`

DISCON_PTS_AUDIO:

5: `"audio" System Load discontinuity - dropping audio due to system or disk load audio = %d, video = %d, chan = %d\n", "PTS discontinuity - first audio after video, drop video = %llu, %llu, %llu, %d\n", "PTS discontinuity - audio dts gap to last = %llu, %llu, %d, %llu, %d\n", "PTS discontinuity - audio dts gap to next = %llu, %llu, %llu, %d, %llu, %d\n"`

6: `"audio" "[Extra %d = %f samples / %f samples per frame][Last Audio pts: %f Current: %f (%f) PerPacket %f Last Good Audio %ld] Last Audio Pts[%ld %ld %ld %ld] AudioPackets[%d %d %d %d] Last Video PTS %d", "System Load discontinuity - dropping audio due to system or disk load [video Q %d] [audio Q %d-%d-%d-%d-%d-%d-%d-%d] current audio chan = %d Audio Handler %08X # of channels %d", "PTS discontinuity - first audio after video, drop video. video pts %llu, check end %llu chan %d V{%llu - %llu} A{%llu - %llu}", "PTS discontinuity - audio dts gap to last = %llu, %llu, %d, %llu, %d", "PTS discontinuity - audio dts gap to next = %llu, %llu, %llu, %d, %llu, %d", "Audio discontinuity - No Audio found %d Video Q(%d) %llu-%llu Audio Q(%d) %llu-%llu", "PTS discontinuity - audio dts gap %llu last %llu",`

4.4 Error Returns

#define netxErrorNone 0

No error, success. Normal response.

#define netxErrorUnknown -1

An undefined error.

#define netxErrorBadChannelParam -2
One of the parameters in the channel command is bad.

#define netxErrorClientMissing -3
A command was sent to a client that does not exist.

#define netxErrorStartFailed -4
A start command on a group failed.

#define netxErrorStopFailed -5
A stop command on a group failed.

#define netxErrorRemoveFailed -6
Unable to remove a group or a channel.

#define netxErrorBadGroupParam -7
One of the parameters in the group command is bad.

4.5 Error Returns – Net-X-Copy

0 – Success
1 – unable to create file (EPERM)
2 – no such file or directory (ENOENT).
5 – read or write error (EIO)
6 – no such device or address (ENXIO)
12 – unable to allocate memory (ENOMEM)
22 – invalid argument (EINVAL)
86 – unable to complete copy or conversion.
255 – unable to create index file or unable to index source.
256 – unable to acquire information from source file
257 – unable to set conversion center information
258 – unable to find TS associated with RTIN
259 – unable to retrieve frame information from RTIN
260 – start position in file greater than end position
261 – unable to set target file information
261 – DMF get bytes error
275 – general timecode error (deprecated)
276 – in timecode greater than out timecode
277 – tcin before first timecode in file
278 – timecode is invalid
279 – tcout after last timecode in file

290 – in timecode not in file range
291 – out timecode not in file range
300 – failed to find language codes for audio tracks (set in config)
301 – failed to find audio track channel sources (set in config)
1155 – insufficient permissions to use dmPut

4.6 Error Returns – QC

0 – Success
-1 – General error
-101 – Unable to open the original file
-102 – Unable to open compare file
-103 – Unable to get channel count from the original
-104 – Unable to get channel count from the compare
-105 – Video channel count does not match
-106 – Audio channel count does not match
-107 – Video frame rates do not match
-108 – Audio sample rates do not match
-109 – Stream length does not match
-110 – Video frame length does not match
-111 – Audio sample length does not match
-112 – Starting timecodes do not match
-113 – Timecode types do not match
-114 – Unable to read original video frame
-115 – Original video frame returned zero for size
-116 – Unable to read compare video frame
-117 – Compare video frame returned zero for size
-118 – Closed caption: original file has cc, compare file does not
-119 – Closed caption: compare file has cc, original file does not
-120 – Inner frame read returned error
-121 – Inner frame read has zero size

4.7 Error Returns – Net-X-SDI

These are Microsoft style returns, with error/warning/info bits, and then areas within the DWORD/HRESULT return for source, area and type.

```
#define ERR_SEVERITY_SUCCESS            0x00000000UL
#define ERR_SEVERITY_INFORMATIONAL             0x40000000UL
#define ERR_SEVERITY_WARNING            0x80000000UL
#define ERR_SEVERITY_ERROR              0xC0000000UL

#define ERR_CUSTOMER_CODE               0x20000000UL

#define FACILITY_AVHAL                  0x08010000UL
#define FACILITY_MEDIAFILE              0x08020000UL
#define FACILITY_VVW                    0x08030000UL
#define FACILITY_VVWEXT                 0x08040000UL
#define FACILITY_VVWINT                 0x08050000UL
#define FACILITY_VVWNET                 0x08060000UL
#define FACILITY_VVWCTL                 0x08070000UL
#define FACILITY_TCS                    0x08080000UL
#define FACILITY_CLIP                   0x08090000UL
#define FACILITY_UTIL                   0x080A0000UL
#define FACILITY_MEM                    0x08100000UL
#define FACILITY_MEDIABASE              0x08110000UL
#define FACILITY_DRASTIC                0x0FFF0000UL

#define DEFAULT_FACILITY                FACILITY_DRASTIC

//
// Generic sub areas for errors to occur
//
#define VVWSUB_HANDLE                           0x0000
#define VVWSUB_CHANNEL                          0x1000
#define VVWSUB_MEMORY                           0x2000
#define VVWSUB_FILE                             0x3000
#define VVWSUB_READ                             0x4000
#define VVWSUB_WRITE                            0x5000
#define VVWSUB_HARDWARE                         0x6000
#define VVWSUB_MEDIACMD                         0x7000
#define VVWSUB_DFRAME                           0x8000

#define VVWSUB_INTERNAL                         0xE000
#define VVWSUB_SOFTWARE                         0xF000

//
// Specific errors
```

```
//
enum {
	VVWERR_DLIST,		// Handling dlist
	0		0x00
	VVWERR_DOESNTEXIST,	// Handles, pointers, drivers
	VVWERR_UNKNOWN,		// Channel
	VVWERR_INVALID,		// Handles, pointers
	VVWERR_BAD,		// Handles, pointers, mediacmd, dframe
	VVWERR_NULL,		// Pointer
	VVWERR_SELECTION,	// Channel select, in/out, clip
	VVWERR_LOCAL,		// Memory
	VVWERR_GLOBAL,		// Memory
	VVWERR_SHARED,		// Memory, resource
	VVWERR_FREE,		// Memory, resource
	10		0x0A
	VVWERR_TOOSMALL,	// Memory area too small
	VVWERR_OPEN,		// File open
	VVWERR_END,		// End of File
	VVWERR_READ,		// File, memory
	VVWERR_WRITE,		// File, memory
	VVWERR_CLOSE,		// File
		16		0x10
	VVWERR_NOTFOUND,	// File, resource
	VVWERR_NOTOPEN,		// File
	VVWERR_PARAM,		// Calling
	VVWERR_PARAM1,		// Calling
	20
	VVWERR_PARAM2,		// Calling
	VVWERR_PARAM3,		// Calling
	VVWERR_PARAM4,		// Calling
	VVWERR_PARAM5,		// Calling
	VVWERR_INDIRECTION,	// Recursion
	VVWERR_ADD,		// Adding to list, tcspace, clipspace
	VVWERR_DEL,		// Deleting/Removing from list, tcspace, clipspace
	VVWERR_FIND,		// Finding
	VVWERR_CALLBACK,	// User error
	VVWERR_FILEEXISTS,	// Creating existing file                                30
	VVWERR_LOCKED,		// Lock exclusive by someone else
	VVWERR_NOALLOC,		// No Allocation available
	0x20
	VVWERR_EXCEPTION,	// Caused an exception
	VVWERR_SEEK,		// Unable to seek
	VVWERR_NOT_READY,	// File has not yet finished write
	VVWERR_NOT_ENOUGH_DATA,	// File has not yet finished write
	VVWERR_NOT_SUPPORTED,	// Not a supported operation
	VVWERR_TOLARGE,		// Too many too big
	VVWERR_NONEAVAIL,	// A request param was not available
	VVWERR_BAD_FORMAT,	// Item format not correct                               40
```

```
        0x28
        VVWERR_OUTOFRANGE,
// Sentinal
        __VVWERR_SENTINAL = 0xFFF  // last possible error
};
#define VVWERR_MASK  0xFFFFUL
```

4.8 ACK/ACKC/ACKR File Format

The ACK(R)/Acknowledge file is created after a file is converted, clipped or captured with Net-X-Code/Net-X-Copy. This file will be created when the file it is associated with is complete and closed. It is an XML formatted file that includes information about the output file and its source, and is only created once the output file is completely ready. Here is a sample file:

<?xml version='1.0' encoding='ISO-8859-1'?>
<ACKNOWLEDGE>
 <DETAILS>
 <FILE NAME="NetXCode Capture" DSTNAME="47448317-983f-4b35-9ba5-52265522a9ba_HBR.mxf">
 <RESULT>0</RESULT>
 <COMMENT>
Packets -> Total=546095529 [missing=2], Video=266125460 [dis=0], Audio={29866438,29866438,29866438,29866439,0,0} [dis=0], Dolby=0 [dis=0], Tc=447550 [dis=0], MetaData=23087 [dis=0], CC=1790200 [dis=0] </COMMENT>
 <SRCPATH>NetXCode Capture</SRCPATH>
 <DSTPATH>/bass3fs/share/Ingest/recordings/2017/06/20/47448317-983f-4b35-9ba5-52265522a9ba_HBR/47448317-983f-4b35-9ba5-52265522a9ba_HBR/47448317-983f-4b35-9ba5-52265522a9ba_HBR.mxf</DSTPATH>
 <RTINPATH>/bass3fs/share/Ingest/recordings/2017/06/20/47448317-983f-4b35-9ba5-52265522a9ba_HBR/47448317-983f-4b35-9ba5-52265522a9ba_HBR/extra/drastictech.com/47448317-983f-4b35-9ba5-52265522a9ba_HBR.rtin</RTINPATH>
 <MCCPATH>/bass3fs/share/Ingest/recordings/2017/06/20/47448317-983f-4b35-9ba5-52265522a9ba_HBR/47448317-983f-4b35-9ba5-52265522a9ba_HBR/extra/drastictech.com/47448317-983f-4b35-9ba5-52265522a9ba_HBR_vanc.mcc</MCCPATH>
 <ACKPATH>/bass3fs/share/Ingest/recordings/2017/06/20/47448317-983f-4b35-9ba5-52265522a9ba_HBR/47448317-983f-4b35-9ba5-52265522a9ba_HBR/extra/drastictech.com/47448317-983f-4b35-9ba5-52265522a9ba_HBR.ack</ACKPATH>
 <TIME>Wed Jun 21 00:05:01 2017
 </TIME>
 <INFO>
  <General>
   <Format>MXF AS-02</Format>

```
            <Duration>871908</Duration>
          </General>
          <Video>
            <Format>h.264 AVC1</Format>
            <FourCC>0x61766331</FourCC>
            <Width>1280</Width>
            <Height>720</Height>
            <BitCount>24</BitCount>
            <Rate>60000</Rate>
            <Scale>1001</Scale>
            <Length>871908</Length>
          </Video>
          <Audio>
            <Format>AES-3</Format>
            <Channels>8</Channels>
            <Frequency>48000</Frequency>
            <Bits>24</Bits>
            <Length>698223926</Length>
          </Audio>
          <Metadata>
            <Timecode>23:55:04;46 d</Timecode>
            <UserBits>10702600</UserBits>
            <VITCTimecode>23:55:04;46 d</VITCTimecode>
            <VITCUserBits>10702600</VITCUserBits>
            <UMID>828392c0-8539-4f81-bab4-b243d2771232</UMID>
            <PosterFrame>0</PosterFrame>
            <A-Frame>0</A-Frame>
          </Metadata>
        </INFO>
        <DISCONTINUITIES>
    <Discontinuity type="timecode" time="20:00:05 Tuesday, June 20, 2017" frame="17866"
LastITimeCode="318" DisconTimeCode="332" timecodetype="d" comment="CurrentTimeCode:
327">00:00:05;32</Discontinuity>
    <Discontinuity type="timecode" time="21:24:11 Tuesday, June 20, 2017" frame="320312"
LastITimeCode="302757" DisconTimeCode="302771" timecodetype="d"
comment="CurrentTimeCode: 302770">01:24:11;15</Discontinuity>
    <Discontinuity type="timecode" time="21:24:11 Tuesday, June 20, 2017" frame="320327"
LastITimeCode="302770" DisconTimeCode="302784" timecodetype="d"
comment="CurrentTimeCode: 302787">01:24:11;28</Discontinuity>
        </DISCONTINUITIES>
      </FILE>
  </DETAILS>
</ACKNOWLEDGE>
```

ACKR file sections

- FILE Name – includes the source file and the destination file name
- RESULT – one of the results, either Net-X-Code or Net-X-Copy above
- SRCPATH – full source path
- DSTPATH – full destination path
- RTINPATH – output rtindex path
- ACKPATH – output ack path
- TIME – the time the operation completed
- DISCONTINUITIES – any time gaps in the capture. Types included
  - “video”
  - “audio”
  - “dolby”
  - “timecode”
  - “meta tc”
  - “meta cc”
  - “metadata”
  - “audio pts”
  - “video pts”
- INFO – This section includes information about the output file

General

- Format – the main format of the output file (MXF/MOV/MP4/etc)
- Duration – the total number of frames in a file

Video

- Format – the video format (MPEG-2/XAVC/etc)
- FourCC – the four character code as a hex value
- Width – the video width
- Height – the video height
- BitCount – the video bit depth
- Rate – the video rate (rate/scale = fps)
- Scale – the video scale (rate/scale = fps)
- Length – the length of the video stream in frames

Audio

- Format – the audio format (AAC/AC-3/etc)
- Channels – total number of channels containing audio
- Frequency – the audio frequency
- Bits – the bit size of an audio sample
- Length – length of the audio stream in audio samples

MetaData

- Any available metadata per the Drastic metadata XML spec

4.9 Discontinuity Handling in Net-X-Copy

Net-X-Copy will skip over short discontinuities and grab the next available frame. This will cause a little hiccup in the output file.

For larger discontinuities the copy will fail and report the time code of the frame it was trying to read in the comment section of the ack.

4.10 Configuration

The configuration files/settings are stored in different places for different operating systems:

Windows:
Registry
\HKEY_CURRENT_USER\Software\Drastic\NetXBase
\HKEY_CURRENT_USER\Software\Drastic\NetXCmd
\HKEY_CURRENT_USER\Software\Drastic\NetXCmd\Groups
\HKEY_CURRENT_USER\Software\Drastic\NetXCopy
\HKEY_CURRENT_USER\Software\Drastic\NetXTimecode

Linux:
~/.config/Drastic/NetXBase.conf
~/.config/Drastic/NetXCmd.conf
~/.config/Drastic/NetXCopy.conf
~/.config/Drastic/NetXTimecode.conf

macOS:
${HOME}/Library/Preferences/com.drastic.NetXBase.plist
${HOME}/Library/Preferences/com.drastic.NetXCmd.plist
${HOME}/Library/Preferences/com.drastic.NetXCopy.plist
${HOME}/Library/Preferences/com.drastic.NetXTimecode.plist

4.10.1 Net-X-Base (NetXCode)

Geometry – string – position of UI
forcegui – dword – optional, force the GUI on (1) or off (0)
commandip – string – IP address of the interface for Net-X-Cmd/Base to communicate on
multicast – string – the multicast address for Net-X-Base and Net-X-Cmd to use to find each other (default 230.7.7.7)

4.10.2 Net-X-Cmd

Geometry – string – position of UI
forcegui – dword – optional, force the GUI on (1) or off (0)
commandip – string – IP address of the interface for Net-X-Cmd/Base to communicate on
videoip – string – IP address of the interface for Net-X-Code to capture network video streams on
autostart – if true, start capture of any enabled groups as soon as Net-X-Cmd starts up
tod – bool – use time of day
fileCollisionAction – int – what to do if the file already exists
    1 – fail the capture
    2 – rename the new file to continue the capture
    3 – append if possible, otherwise rename

\Groups\
  NetXCode### ← each group of channels
    autostart – if true, start capture of this group as soon as Net-X-Cmd starts up
    keyname – the name that can be used to identify the group for commands
    name – the friendly name of the group
    version – the version of the software that created this group
    Channel### - each channel
      address – string – IP address of rtp/udp sender
      directory – string – where to record the files
      enabled – whether this channel is enabled or not
      mp4proxydatarate – target data rate when converting inline to mp4
      mp4proxyfcc – the mp4 codec to use for the proxy
      mp4proxymode – if 1 then use the proxy/downconvert
      mp4proxyscaleddown – the factor to scale the proxy by (2=1/2, 3=1/3, 4=1/4)
      name – string – base name of the file to record to
      port – string - RTP/UDP port
      protocol – string – RTP or UDP
      type – string – type of file to record: ts, ts-hls, mxf, mxf-as02, mxf-xdcam, mxf-open, mp4, mp4-fmp4, mp4-dash, mov

4.10.3 Net-X-Copy

geometry – string – position of UI
forcegui – dword – optional, force the GUI on (1) or off (0)
startnetx – dword – start netx

4.10.4 Net-X-Time-Code

address – group address of the time code stream
port – port for the time code stream
protocol – currently UDP or RTP

fixedpreview – for testing
name – for testing – file source
type – for testing

4.11 General Config: config.xml

The config.xml file contains the general configuration for the system, including codec, plugin and all the SDI channel information. This file can be found here:

4.11.1 Windows:

C:\ProgramData\Drastic\config.xml
Alt - C:\Documents and Settings\Drastic\config.xml

4.11.2 Linux:

/etc/Drastic/config.xml
Alt - /Home//Drastic/config.xml

4.11.3 macOS:

/Library/Applications Support/Drastic/config.xml
Alt - /Home//Drastic/config.xml

Generally, the alt location is for users without administrator/super-user credentials. The ideal location is in the system area, so it is shared by all users, but per user config is also supported for machines that are locked down.

4.12 Automatic Proxy Generation Setup

Net-X-Code can create a low res proxy for streams captured from IP or SDI. Normally the proxy will be an MP4, but various types of multiple bitrate files are also supported as MP4, HLS/TS and DASH fMP4. To set up the proxy that will be generated for every stream that is recorded or converted, you will need to modify **config.xml**, in the location described above.
In config.xml, find the area
<DrasticConfig>
<VVW>
<Internal0>
Within the <Internal0> tag, there are four elements that control the automatic proxy:
<proxymode Value="1" Type="2" />
0 – no automatic proxy
1 – create proxy on record/write
2 – create proxy on playback
3 – create proxy on read and write

<proxyfiletype Value="211" Type="2" />
197 – create MP4 proxies
210 – create multirate HLS/TS proxies
211 – create multirate MP4-Dash proxies
(note: multirate proxies can have just one rate/size/etc)

<proxycompressiontype Value="1635148593" Type="2" />
1635148593 – (avc1) h.264 compression

<proxydatarate Value="3000" Type="2" />
3000 – 3 megabits. Setting is in kilobits.

To setup one or more multirate TS/HLS or MP4/DASH proxies, see the multirate setup below.

4.13 Multirate TS/HLS and MP4/DASH setup

The multirate configuration is contained in the **config.xml** file, which can be located with the information described above. There is only one multirate setup, which will be applied to HLS or DASH, depending on what file is specified to be created. The configuration keys can be found here:
<DrasticConfig>
<MediaReactor>
<PlugIns>
<MultiRate>
The first setting sets how many files will be created. Under the MultiRate key, there will be
<TotalMultiRateFiles Value=”3” Type=”2” />
The value contains the number of multirate files to create. In this case files are being recorded. Each

multirate file will require a sub key of File# (eg, <File0>, <File1>, <File2>) under the <MultiRate> key. Within each <File#> key, the following settings are supported:

<Width Value="512" Type="2" />
The width of the file to be created. Should be a multiple of 8.

<Height Value="288" Type="2" />
The height to be created. Should be a multiple of 16, or a standard value like 1080

<BitRateAvg Value="3000" Type="2" />
The target average bit rate in kilobits per second

<BitRatePeak Value="3200" Type="2" />
The peak bit rate in kilobits per second

<Profile Value="77" Type="2" />
The h.264 profile to use while encoding (optional)

<Level Value="30" Type="2" />
The h.264 level to use while recording (optional)

4.14 System Setup

Net-X-Code makes use of a number of TCP and UDP ports for discovery, connection and capture. All of these connections must be allowed to pass through any firewall or other network protection for Net-X-Code to work. The main ports Net-X-Code uses include:

80/443 – TCP - Apache server for standard HTML and Net-X-Player
20/21 – TCP - Optional vsFtp for file access

7630 – TCP – Net-X-Cmd/Net-X-Code server command port
57500-57XXX – UDP = Communications server port
58500 – UDP – Net-X-Base multicast server port
58500 – TCP – Net-X-Cmd→Net-X-Base communication port (outgoing)
59000->50### – TCP – Net-X-Base<-Net-X-Cmd communications port, where ### is the max number of channels (incoming)

By default, Net-X-Base uses this multicast address to join all of its components. Please note, this can be changed in the configuration to allow multiple Net-X-Code groups to exist on the same network.

230.7.7.7 – Net-X-Code system discovery multicast address

The individual streams being captured or transmitted also use multicast or unicast addresses, along with UDP or TCP ports. These are user configured, but by default, RTP and UDP traffic often use port 5004 (default for RTP) or port 1234 (the experimental port).

The basic connection process for the whole system is:

- Net-X-Base hosts multicast at 230.7.7.7:58500 UDP for Net-X-Cmds to connect to
- Net-X-Cmd joins 230.7.7.7:(57500 +offset) and sends a message to Net-X-Base
- Net-X-Base responds back through multicast
- Net-X-Cmd receives the message and gets the Net-X-Base IP from the messages
- Net-X-Cmd makes a TCP connect from Net-X-Cmd <ip>:58500 to Net-X-Base <baseip>:59000-50###
- Net-X-Code is spawned, it connects back to the local Net-X-Cmd on port 7630

4.14.1 Linux (CentOS/Red Hat) - Network

If you are in a protected network, the simplest way to get everything working is to disable the iptables firewall (Note: you'll need to be root for all of this)

/etc/init.d/iptables stop

This is also a good way to test that Net-X-Code is working properly on your system. For most systems, iptables will be required, so individual exceptions should be added for each port/type on each adapter. Iptables can then be turned on and off to check that all the correct ports are being allowed through. For the multicast parts rp_filter setting must be changed to be more permissive:

Open /etc/sysctl.conf
Change the line:
net.ipv4.conf.default.rp_filter = 1
to:
net.ipv4.conf.default.rp_filter = 2

This setting may be set for individual ethernet devices so make sure to change net.ipv4.conf.eth0.rp_filter as well.

4.14.2 Windows - Network

If you run each of the components from the desktop, the Windows OS/firewall will ask if you want to add an exception for each of the applications (netxbase, netxcmd, netxcode, netxcopy) as they start up and connect to the network. Once you have allowed these exceptions, they can be run

remotely/headless, and will work properly.

4.14.3 macOS - Network

To disable the firewall for Net-X-Code, for each application:

1. Open System Preferences > Security & Privacy > Firewall > Firewall Options.
2. Click Add.
3. Choose an application from the Applications folder and click Add.
4. Ensure that the option next to the application is set to Allow incoming connections.
5. Click OK.

4.14.4 Linux – Required Packages

SDL

yum install SDL

libstatgrab

yum install libstatgrab-devel

libuuid

yum install e2fsprogs-devel

libgstreamer

yum install gstreamer gst-plugins-base

wxGTK – only for legacy apps

yum install wxGTK

4.14.5 Linux – SysLog Output

To set up syslog output, add the following lines to /etc/rsyslog.conf:

```
#route all dt messages to custom log
:app-name, isequal, "dtlog"                    /var/log/dtlog.log
```

and then create that file

```
sudo touch /var/log/dtlog.log
```

5 Copyrights and Trademark Notices

5.1 General

Copyright 2025, Drastic Technologies Ltd. All rights reserved worldwide. No part of this publication may be reproduced, transmitted, transcribed, altered, or translated into any languages without the written permission of Drastic Technologies. Information and specifications in this document are subject to change without notice and do not represent a commitment on the part of Drastic Technologies.

A&E Television Networks - A&E Networks is a trademark of A&E Television Networks

Adobe, Inc. - Adobe, the Adobe logo, Adobe Premiere, Adobe After Effects, Creative Cloud, Frame.io, and Iridas are either registered trademarks or trademarks of Adobe in the United States and/or other countries.

Advanced Micro Devices, Inc. - AMD is a trademark of Advanced Micro Devices, Inc.

ADVANTECH CO., LTD - ADVANTECH and B&B are trademarks of ADVANTECH CO., LTD

AES Audio Engineering Society - AES and Audio Engineering Society are trademarks of the Audio Engineering Society

aescripts + aeplugins - ZXPInstaller Copyright aescripts + aeplugins 2023

AIMS Alliance - The AIMS Alliance is a trademark of Alliance for IP Media Solutions (AIMS).

AJA Video Systems, Inc. - AJA® is a registered trademark of AJA Video Systems, Inc. AJA™ is a trademark of AJA Video Systems, Inc. Corvid Ultra®, KONA®, IO®, KUMO®, U-Tap®, and T-Tap® are registered trademarks of AJA Video Systems, Inc.

Amazon Web Services, Inc. - Amazon, AWS and Smile Logo, Powered by AWS Logo, AWS Co-Marketing Tools, the Partner Logo, the Program Marks, Amazon Web Services, AWS, AWS S3, and the names of AWS products, services, programs, and initiatives are trademarks or registered trademarks of Amazon Web Services, Inc.

Amberfin Limited - AMBERFIN is a trademark of Amberfin Limited.

AMERICAN BROADCASTING COMPANIES, INC - ABC is a trademark of AMERICAN BROADCASTING COMPANIES, INC

American Cinematographer - The ASC, American Cinematographer and Friends of the ASC are trademarks of the American Society of Cinematographers. (All rights reserved)

AMWA Advanced Media Workflow Association, Inc. - Copyright © 2025 AMWA – Advanced Media Workflow Association. All rights reserved.

Animation Magazine - © 2025 Animation Magazine. All Rights Reserved. The Business, Technology & Art Of Animation And VFX

Apple Inc. - Apple, the Apple logo, Final Cut, Final Cut Pro, Apple TV, iOS, iPad, iPhone, iPod touch, iTunes, Mac, Mac OS X, macOS, Shake, Final Cut Pro, ProRes, High Sierra, Mojave, Ventura, Sonoma, M1, M2, and QuickTime are trademarks of Apple Inc., registered in the U.S. and other countries. OpenCL and the OpenCL logo™ are trademarks owned by Apple Inc. and licensed to the Khronos Group.

ARRI AG – ARRI, Arri T-Link, and Alexa are registered trademarks of the ARRI Group

ASSIMILATE® Inc. - Assimilate SCRATCH and Assimilate SCRATCH Lab are either trademarks

or registered trademarks of ASSIMILATE® Inc. or its subsidiaries in the United States and/or other countries.

ATI TECHNOLOGIES ULC - ATI is a trademark of ATI TECHNOLOGIES ULC

ATSC: The Broadcast Standards Association - © 2025 ATSC Advanced Television Systems Committee, Inc.

Autodesk, Inc. - Autodesk, Discreet, Flame, Flare, Smoke, Lustre, Maya, and Moxion are either trademarks or registered trademarks of Autodesk, Inc. or its subsidiaries in the United States and/or other countries.

Avid Technology, Inc. - Avid Media Composer®, Avid MediaCentral®, Avid Interplay®, ProTools®, and Avid NewsCutter® are either trademarks or registered trademarks of Avid Technology, Inc. or its subsidiaries in the United States and/or other countries.

Axis Communications AB - Axis is a registered trademark of Axis Communications AB

Bell Media Inc. - Bell Media, BNN, CP24, CTV, CTV TWO, Much, MuchMusic and The Comedy Network, and all associated designs and logos are trademarks of Bell Media Inc.

Belle Nuit Montage - Matthias Bürcher August 2000-2016. All rights reserved. Written in Switzerland. Starting 2016 Belle Nuit Subtitler is released under the GNU Lesser General Public License

BirdDog Software Corporation - BIRDDOG is a trademark of BirdDog Software Corporation

Blackmagic Design Pty. Ltd. - DaVinci Resolve, DaVinci Fusion, UltraStudio, DeckLink, Intensity Pro 4K, UltraScope, and RED are either trademarks or registered trademarks of Blackmagic Design Pty. Ltd. or its subsidiaries in the United States and/or other countries.

Bluefish Technologies - Bluefish444, IngeSTore, Symmetry, Kronos, Epoch, Epoch:Neutron, Fury, Lust, Vengeance HD, Deepblue, Envy SD, and Epoch:SuperNova are trademarks of Bluefish Technologies

Boris FX, Inc. - Boris FX, Sapphire, and Silhouette are trademarks of Boris FX, Inc.

Bridge Digital, Inc. - Bridge Digital is a trademark of Bridge Digital, Inc..

Bridge Technologies Co AS - Bridge Technologies is a trademark of Bridge Technologies Co AS

Bright Technologies, Inc. - Bright and Bright Systems are trademarks of Bright Technologies, Inc.

British Broadcasting Corporation - BBC is a trademark of British Broadcasting Corporation

Broadcast Beat - © 2025 Relevant Media Properties, LLC. All Rights Reserved.

BT Group plc - BT is a trademark of BT Group plc

Cable News Network, Inc. - CNN is a trademark of Cable News Network, Inc.

Canadian Federal Institutions - Official symbols of federal institutions, including the Arms of Canada may not be reproduced, whether for commercial or non-commercial purposes, without prior written authorization.

CANON KABUSHIKI KAISHA - CANON is a trademark of CANON KABUSHIKI KAISHA

Catapult Group International Ltd - Catapult is a trademark owned by Catapult Group International Ltd

Changsha Kiloview Electronics Co., Ltd - KILOVIEW is a trademark of Changsha Kiloview Electronics Co., Ltd

Charter Communications Inc. - Charter Communications is a trademark of Charter

Communications Inc.

CineSys LLC – CineSys is a registered trademark of CineSys LLC.

Cisco Systems, Inc. - Cisco, and Webex are registered trademarks of Cisco Systems, Inc.

Cloudfirst Technology Solutions Inc. - Cloudfirst is a registered trademark of Cloudfirst Technology Solutions Inc.

Cobalt Digital - Cobalt Digital is a registered trademark of Cobalt Digital Inc.

Codex Corporation - CODEX and Action Cam are trademarks of Codex Corporation

Comcast Corporation - Sky UK Limited is a wholly owned subsidiary of Comcast Corporation

Comtrol Corporation - Comtrol is a registered trademark of Comtrol Corporation

CoreCodec, Inc. - MATROSKA is a trademark of CoreCodec, Inc.

Corel Corporation - WinZip, the WinZip vise and file logo, and Pinnacle are registered trademarks of Corel Corporation

CORSAIR MEMORY, INC. - ELGATO is a trademark of CORSAIR MEMORY, INC.

Corus Entertainment Inc. - CORUS is a trademark of Corus Entertainment Inc.

Crayon Software Experts Spain SL - Crayon is a trademark of Crayon Software Experts Spain SL

CrypKey Inc (formerly Kenonics) - CrypKey is a registered trademark of CrypKey Inc.

Deadline - Deadline is a part of Penske Media Corporation. © 2025 Deadline Hollywood, LLC. All Rights Reserved.

Deltacast - © Copyright 2024 DELTACAST. All rights reserved

Deluxe Media Inc. - Deluxe is a trademark of Deluxe Media Inc.

Digital Formation, Inc. - Digital Formation is a Copyright of Digital Formation, Inc.

Digital Video Systems Ltd - DVS is a trademark of Digital Video Systems Ltd

DIGITNOW! - Digitnow is a trademark of DIGITNOW!

Docker Inc. - DOCKER is a trademark of Docker, Inc.

Dolby Laboratories – Dolby, Dolby Vision, the double-D symbol, and Millicast are registered trademarks of Dolby Laboratories.

DPP - The Digital Production Partnership - DPP is a registered trademark | Digital Production Partnership © 2025

Drastic Technologies, Ltd. – 2110Scope, 4KScope, ccConvert, Drastic Technologies, DrasticPreview, DrasticScope, FlowCaster, HDRScope, Media File Scanner, MediaNXS, MediaReactor, MediaReactor Workstation, MR Lite, ndiScope, Net-X-Code Channel, Net-X-Code Server, Net-X-Convert, Net-X-Proxy, Network Video Analyzer, NetXfer, NETXROUTER, NetXScope, QuickClip, sdiScope, SyncControl, TcCalc, TestPatternGenerator, videoQC Inspect, videoQC Pro, videoQC View, and videoQC Workstation are trademarks of Drastic Technologies Ltd.

DTS - DTS, the Symbol, and DTS and the Symbol together are registered trademarks of DTS, Inc.

Dublin Core™ Metadata Initiative - "Dublin Core" is a protected under common law trademark of the Dublin Core™ Metadata Initiative.

Eastman Kodak Company - Cineon™ is a trademark of Eastman Kodak Company

Eaton Corporation plc - Eaton, Tripp Lite, and PowerAlert are registered trademarks of Eaton Corporation plc

EBU - Copyright EBU 2025. All rights reserved.

Empress Media Asset Management (eMAM) – eMAM, and eMAMDirector are registered trademarks of Empress Media Asset Management (eMAM)

Entertainment and Sports Programming Network - ESPN is a trademark of Entertainment and Sports Programming Network

Epic Games, Inc. - UNREAL ENGINE is a trademark of Epic Games, Inc..

Epiphan - All Epiphan product names and logos are trademarks or registered trademarks of Epiphan

Evercast, LLC - EVERCAST is a trademark owned by Evercast, LLC

Evertz Technologies Limited - Evertz is a registered trademark of Evertz Technologies Limited

EVS Broadcast Equipment - EVS is a registered trademark of EVS Broadcast Equipment

Fabrice Bellard - FFmpeg is a trademark of Fabrice Bellard

Filestage GmbH - Filestage is a trademark of Filestage GmbH

FilmLight Ltd. - FilmLight and BaseLight are trademarks of FilmLight Ltd.

Filmworkz - Filmworkz is an operating brand of BlissTek Ltd. BlissTek Ltd. Filmworkz Nucoda is either a trademark or registered trademark of BlissTek Ltd. or its subsidiaries in England, Wales, and/or other countries.

For-A - For-A is a registered trademark of FOR-A COMPANY LIMITED, Copyright © FOR-A Company Limited.

France Télévisions - France.tv is a trademark of France Télévisions

Fraunhofer IIS and Thomson Multimedia - MPEG Layer-3 audio coding technology licensed from Fraunhofer IIS and Thomson Multimedia.

Fraunhofer-Gesellschaft zur Förderung deer angewandten Forschung e.V. - EASYDCP is a trademark and brand of Fraunhofer-Gesellschaft zur Förderung deer angewandten Forschung e.V..

Free Software Foundation (FSF) - Portions of this product are licensed under LGPL, governed by the GNU LESSER GENERAL PUBLIC LICENSE, published by the Free Software Foundation (FSF).

Ftrack AB - FTRACK is a trademark and brand of Ftrack AB

Gen Digital Inc. (formerly Symantec Corporation and NortonLifeLock) - Symantec, Symantec Endpoint Virtualization Suite, Sygate, Altiris, and Altiris Virtualization Agent are registered trademarks of Gen Digital Inc.

Google LLC – YouTube, Google, Google Cloud, Google.meet.com, and Android are registered trademarks of Google LLC

GoPro, Inc. - Cineform® is a trademark or registered trademark of GoPro, Inc.

Grass Valley USA, LLC - Grass Valley®, GV®, the Grass Valley logo, and EDIUS® are trademarks or registered trademarks of Grass Valley USA, LLC, or its affiliated companies in the United States and other jurisdictions.

HaiVision Systems, Inc. - Haivision is a registered trademark of HaiVision Systems, Inc.

Harmonic - Harmonic is a registered trademark of Harmonic Inc.

Harris Corporation - Harris, and Leitch Technology Corp. are registered trademarks of Harris Corporation

Hewlett Packard Enterprise Company – OpenGL and SGI are registered trademarks and the

OpenGL SC logo is a trademark of Hewlett Packard Enterprise Company

Hewlett Packard Group LLC - HP is a trademark of HP Hewlett Packard Group LLC.

i-scream - i-scream is a trademark of i-scream

IABM - © 2025 IABM IABM is company limited by guarantee. Registered in England No: 5262009. Registered Office: IABM, 5 Deansway, Worcester, WR1 2JG

IBC - IBC (International Broadcasting Convention) is owned and run by the IBC Partnership, comprising six industry bodies: IEEE, IET, IABM, SCTE, SMPTE, and RTS.

Ideal Systems Asia Pacific Ltd. - Ideal Systems is a registered trademark of Ideal Systems Asia Pacific Ltd.

IEEE - IEEE Broadcast Technology Society - The IEEE emblem is a trademark owned by the IEEE for the purpose of indicating membership in the IEEE.

Ikegami Electronics (USA) Inc. - EditCam is a registered trademark of Ikegami Electronics (USA) Inc.

Indiecam GmbH - IndieCam is a registered trademark of Indiecam GmbH

Infocomm - InfoComm, AVIXA and associated logos are a trademark or registered trademark of AVIXA

INOGENI Inc - INOGENI® is a Registered Trademark and TOGGLE is a Trademark of INOGENI Inc

Institute of Electrical and Electronics Engineers - IRE is a trademark of the Institute of Electrical and Electronics Engineers

INTEL CORPORATION - INTEL is a trademark of INTEL CORPORATION

International Business Machines Corporation (“IBM”) - IBM® is a trademark owned by International Business Machines Corporation (“IBM”) and might also be trademarked or a registered trademark in other countries

Interactive Effects, Inc. - Piranha is a registered trademark of Interactive Effects, Inc.

Intraware, Inc. – Intraware is a registered trademark of Intraware, Inc.

IO Industries Ltd. - IO Industries is a trademark of IO Industries Ltd.

Iteris, Inc. - Odetics is a registered trademark of Iteris, Inc.

JVC KENWOOD CORPORATION - JVC is a trademark of JVC KENWOOD CORPORATION

Kinefinity Inc. - KINEFINITY is a trademark of Kinefinity Inc.

L3Harris Technologies, Inc. - Louth is a trademark of L3Harris Technologies, Inc.

LeeLu Soft - Watch 4 Folder is a trademark of LeeLu Soft

LinkedIn Corporation - LinkedIn is a trademark of LinkedIn Corporation

Linus Torvalds - Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

Logitech International SA - LOGITECH is a trademark of Logitech International SA

LogMeIn, Inc. - GoTo is a trademarks and service marks of LogMeIn, Inc., and may be registered in the U.S. Patent and Trademark Office and in other countries.

Louper.io Ltd - Louper.io is a trademark of Louper.io Ltd

Lynx Technik AG - LYNX TECHNIK AG is a trademark of LYNX TECHNIK AG.

Magic Lantern - Magic Lantern is a registered trademark of Magic Lantern

MAINCONCEPT GMBH - MAIN CONCEPT is a trademark of MAINCONCEPT GMBH

Marshall Electronics, Inc. - Marshall is a registered trademark of Marshall Electronics, Inc.

Mastercard International Incorporated - Mastercard is a trademark of Mastercard International Incorporated

Matrox Electronic Systems, Ltd - Matrox and Matrox product names are registered trademarks and/or trademarks of Matrox Electronic Systems, Ltd.

MediaArea.net SARL - MediaInfo - Copyright © 2002-2013 MediaArea.net SARL. All rights reserved.

Mellanox Technologies, Inc - Mellanox® and ConnectX® are registered trademarks of Mellanox Technologies, Inc

Meta Platforms, Inc - Facebook and Instagram are trademarks of Meta Platforms, Inc

Metro-Goldwyn-Mayer Studios, Inc. - Metro Goldwyn Mayer, and MGM, are trademarks of Metro-Goldwyn-Mayer Studios, Inc.

Microsoft Corporation – Microsoft: Windows®, Video For Windows (VFW), DirectShow, Microsoft, Skype, Microsoft Azure, Microsoft Teams, Wave Mapper, Microsoft, Windows NT|2000|XP|XP Professional|Server 2003|Server 2008 |Server 2012, Windows 7, Windows 8, Windows 10, Media Player, Media Encoder, Windows Defender, Microsoft Office, .Net, Internet Explorer, SQL Server 2005|2008|2012|2014, Windows Media Technologies and Internet Explorer are trademarks of Microsoft Corporation.

MPEG LA - MPEG LA licenses patent pools covering essential patents required for use of the MPEG-2, MPEG-4, IEEE 1394, VC-1, ATSC, MVC, MPEG-2 Systems, AVC/H.264 and HEVC standards.

Nanjing Magewell Electronics Co. - MagewellTM , ULTRA STREAM® and (the MAGEWELL Logo) are trademarks or registered trademarks of Nanjing Magewell Electronics Co.

National Aeronautics and Space Administration - NASA is a registered trademark of The National Aeronautics and Space Administration

NAB - NABShow and NAB © 2025 National Association of Broadcasters

National Geographic Society - NATIONAL GEOGRAPHIC is a trademark of National Geographic Society

NBA Properties, Inc. - NBA and the NBA logo are trademarks of NBA Properties, Inc.

NBC UNIVERSAL MEDIA, LLC - NBC and NBC Universal are trademarks of NBC UNIVERSAL MEDIA, LLC

Netflix, Inc. - Netflix is a registered trademark of Netflix, Inc.

Nevion - copyright NEVION - All rights reserved. Nevion @ 2023

New Media Manitoba - Copyright © 2025 New Media Manitoba

NewTek, Inc. - NDI, TriCaster, 3Play, TalkShow, Video Toaster, LightWave 3D, and Broadcast Minds are registered trademarks of NewTek, Inc.

Nexidia Inc. - NEXIDIA is a trademark owned by Nexidia Inc.

NGC Corporation - NGC is a registered trademark of NGC Corporation

Nippon Hatsujyo Kabushiki Kaisha - NHK is a trademark of Nippon Hatsujyo Kabushiki Kaisha

Nokia Corporation - OSPREY is a trademark owned by Nokia Corporation

NVIDIA Corporation - NVIDIA, the NVIDIA logo, NVIDIA Quadro, Rivermax, BlueField2, PhysX, and NVIDIA RTX are trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and/or other countries

Object Matrix Limited - ObjectMatrix, and Object Matrix are registered trademarks of Object Matrix Limited

Omneon Video Networks, Inc - Omneon is a trademark of Omneon Video Networks, Inc

ONVIF - the ONVIF primary trademark is the word, “ONVIF”. This trademark has been registered in the United States, European Union, China, Japan and other countries throughout the world.

OpenSSL Project Authors - OpenSSL is a trademark of OpenSSL Project Authors

Oracle Corporation - Oracle®, Java, Front Porch Digital, and MySQL are registered trademarks of Oracle Corporation and/or its affiliates.

Panasonic Holdings Co., Ltd - Panasonic, and Varicam are trademarks of Panasonic Holdings Co., Ltd

Pantone, Inc. - Pantone is a registered trademark of Pantone, Inc

PayPal, Inc. - PAYPAL is a trademark of PayPal, Inc.

PELOTON INTERACTIVE, INC. - PELOTON is a trademark of PELOTON INTERACTIVE, INC.

Pioneer Corporation - Pioneer is a registered trademark of Pioneer Corporation

Post Magazine - © Copyright 2024 Post Magazine. All Rights Reserved.

ProAV - PRO AV SYSTEMS is a trademark of Pro AV Systems, Inc

Production Weekly - Copyright © 2015-2025 Production Weekly

RE:Vision Effects, Inc. - RE:Vision Effects is a registered trademark of RE:Vision Effects, Inc.

Red Hat, Inc. - Red Hat, and the Red Hat logo are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries

Reddit - Reddit's trademarks and other brand assets are owned by Reddit.

Rogers Communications Inc. - Rogers and related marks are trademarks of Rogers Communications Inc. or an affiliate, used under licence.

Ross Video - ©2022 Ross Video Limited, Ross®, MiniME™, and any related marks are trademarks or registered trademarks of Ross Video Limited

Shenzhen Yunlang Technology Co., Ltd. - MOKOSE is a trademark of Shenzhen Yunlang Technology Co., Ltd.

Sigma Design Company, LLC - Sigma Design is a registered trademark of Sigma Design Company, LLC

Sinclair Broadcast Group, Inc. - Sinclair Broadcast Group is a trademark of Sinclair Broadcast Group, Inc.

Snell & Wilcox Limited - SNELL & WILCOX, and Quantel are trademarks owned by Snell & Wilcox Limited

Society of Broadcast Engineers - Copyright, Society of Broadcast Engineers Chapter One, all rights reserved. The SBE logo is used by permission of the Society of. Broadcast Engineers.

Society of Cable Telecommunications Engineers (SCTE) - ©2025 Society of Cable Telecommunications Engineers, Inc. is a subsidiary of CableLabs. All rights reserved.

Society of Motion Picture and Television Engineers - Motion Imaging Journal and SMPTE are trademarks of Society of Motion Picture and Television Engineers.

SoftNI Corporation – SoftNI is a trademark of SoftNI Corporation

Sony Corporation – Sony, Sony DVD Architect, DVD, Catalyst, and Vegas are trademarks of

Sony Corporation and/or its affiliates.

Sound On Sound - copyright © SOS Publications Group and/or its licensors, 1985-2025. All rights reserved.

SRI International - SARNOFF CORPORATION is a trademark of SRI INTERNATIONAL.

SRT (Secure Reliable Transport) - SRT, developed by Haivision, is a royalty free, open source protocol

Streambox Inc. - Streambox is a trademark of Streambox Inc.

Streaming Media - Copyright © 2009 - 2025 Streaming Media Magazine

STREAMWELL LLC – Streamwell is a trademark of STREAMWELL LLC

Technicolor Creative Studios SA - Technicolor is a trademark of Technicolor Creative Studios SA

TechSmith Corporation - CAMTASIA STUDIO is a trademark of TechSmith Corporation

Tektronix, Inc. - Tektronix® and all identified Tektronix trademarks and logos are the property of Tektronix, Inc. or its wholly-owned subsidiaries

Telestream, LLC - Telestream, is a registered trademark, and MacCaption and CaptionMaker are trademarks of Telestream, LLC

The Apache Software Foundation (ASF) - Apache is a registered trademark of The Apache Software Foundation

The Foundry Visionmongers Ltd. - Nuke™ is a trademark of The Foundry Visionmongers Ltd.

The Perl Foundation - Perl and the Perl logo are trademarks of The Perl Foundation

The Qt Company Ltd - The Qt Company Ltd and its subsidiaries (“The Qt Company”) is the owner of Qt trademarks (“Qt trademarks”) worldwide, and “froglogic”, “Squish” and “Coco” are trademarks of the Qt Company Ltd.

THE UNIVISION NETWORK LIMITED PARTNERSHIP - UNIVISION is a trademark of THE UNIVISION NETWORK LIMITED PARTNERSHIP

The Walt Disney Company - Disney, and The Walt Disney Company are trademarks of The Walt Disney Company. LucasFilm is a wholly owned subsidiary of The Walt Disney Company

Toolfarm.com Inc. - Toolfarm is a registered trademark of Toolfarm.com Inc.

Trend Micro Inc. - TrendMicro, and TrendMicro System Protection and registered trademarks of Trend Micro Inc.

Truevision, Inc - TARGA is a registered trademark of Truevision, Inc

TV Asahi Corporation - TV Asahi is a trademark of TV Asahi Corporation

TV Technology - TV Tech is part of Future US Inc, an international media group and leading digital publisher. © Future US, Inc. Full 7th Floor, 130 West 42nd Street, New York, NY 10036.

Twitch Interactive, Inc - TWITCH, the TWITCH Logo, the Glitch Logo, and/or TWITCHTV are trademarks of Twitch Interactive, Inc. or its affiliates.

Twitter, Inc. - Twitter is a wholly owned subsidiary of X Holdings Corp.

Tyler Perry Studios, LLC - Tyler Perry Studios is a trademark of Tyler Perry Studios, LLC

Vefxi Corporation - VEFXi DiamondBlade is a registered trademark of Vefxi Corporation

ViaLA - Via Licensing®, ViaSecure® and the Via logo are registered service marks, and any other Via Licensing names, titles or logos are trademarks or service marks, in each case,

of Via Licensing Corporation, and are protected by law.

Video Clarity, Inc. - Video Clarity and ClearView are trademarks of Video Clarity, Inc.

Video Services Forum - ©2024 Video Services Forum

VideoLAN Non-profit Organization - VideoLAN, VLC, VLC media player and x264 are trademarks internationally registered by the VideoLAN non-profit organization

Videomaker - © Videomaker Inc., 1986 - 2025

Visa International - Visa is a registered trademark of Visa International

Vision Research, Inc - PHANTOM is a trademark of Vision Research, Inc

VITEC - Names and logos identifying products of VITEC are registered trademarks or trademarks of VITEC respectively

Vizrt - VIZRT is a trademark of VIZRT AG.

Warner Bros. Discovery – Discovery, Turner, and Home Box Office, Inc. (HBO), are trademarks of Warner Bros. Discovery

Weisscam GmbH - Weisscam is a trademark and brand of Weisscam GmbH

Wheatstone - ® Wheatstone, Audioarts, and VoxPro are registered trademarks and Wheatstone Layers is a trademark of Wheatstone Corporation

Wizards of OBS, LLC – UNIX, OBS, Open Broadcast Software, the OBS logo, and OBS Studio are trademarks of Wizards of OBS, LLC (The Company)

World Animation Summit - © 2025 Animation Magazine. All Rights Reserved.

World Wrestling Entertainment, Inc. - WWE is a trademark of World Wrestling Entertainment, Inc.

Wowza Media Systems, LLC - Wowza is a trademark of Wowza Media Systems, LLC

wxWidgets - wxWidgets is a trademark of wxWidgets

Xceed Software Inc. - Xceed DataGrid for JavaScript, Xceed Ultimate ListBox for Silverlight, Xceed DataGrid for Silverlight, Xceed DataGrid for WPF, Xceed Grid for .NET, Xceed Zip for .NET, Xceed Real-Time Zip for Silverlight, Xceed Upload for Silverlight, Xceed Zip Compression Library, Xceed FTP for .NET, Xceed Chart for .NET, Xceed Chart for ASP.NET, Xceed SmartUI for .NET, Xceed SmartUI, Xceed Encryption Library, Xceed Binary Encoding Library, Xceed Streaming Compression Library, Xceed Streaming Compression for .NET, Xceed Zip for .NET Compact Framework, Xceed Ultimate Suite, Xceed Data Manipulation Suite, Xceed Absolute Packager are trademarks of Xceed Software Inc.

Xena Networks - Xena is a trademark of Xena Networks

Zapex Technologies - Zapex is a registered trademark of Zapex Technologies

Zhang Haijun - RYBOZEN is a trademark of Zhang Haijun

Ziflow Limited - Ziflow is a trademark of Ziflow Limited

Zixi - Zixi Software and any logos or icons identifying Zixi and the Zixi Software are trademarks of Zixi.

ZLIB - The ZLIB Compressed Data Format Specification is Copyright (C) 1995-2013 Jean-Loup Gailly and Mark Adler.

Zoom Video Communications, Inc. - Zoom and the Zoom logo are trademarks of Zoom Video Communications, Inc.

LGPL: Portions of this product are licensed under LGPL, governed by the following license:

5.2 GNU LESSER GENERAL PUBLIC LICENSE

Version 3, 29 June 2007
Copyright © 2007 Free Software Foundation, Inc. <https://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.
This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.

5.2.1.1 0. Additional Definitions.

As used herein, “this License” refers to version 3 of the GNU Lesser General Public License, and the “GNU GPL” refers to version 3 of the GNU General Public License.
“The Library” refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.
An “Application” is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.
A “Combined Work” is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the “Linked Version”.
The “Minimal Corresponding Source” for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.
The “Corresponding Application Code” for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

5.2.1.2 1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

5.2.1.3 2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

- a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or
- b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

5.2.1.4 3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

- a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the object code with a copy of the GNU GPL and this license document.

5.2.1.5 4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

- a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.
- b) Accompany the Combined Work with a copy of the GNU GPL and this license document.
- c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.
- d) Do one of the following:
  - 0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.
  - 1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the

user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.

- e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5.2.1.6 5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.
- b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

5.2.1.7 6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

Other brands, product names, and company names are trademarks of their respective holders, and are used for identification purpose only.

5.3 MPEG Disclaimers

5.3.1 MPEGLA MPEG2 Patent

ANY USE OF THIS PRODUCT IN ANY MANNER OTHER THAN PERSONAL USE THAT COMPLIES WITH THE MPEG-2 STANDARD FOR ENCODING VIDEO INFORMATION FOR PACKAGED MEDIA IS EXPRESSLY PROHIBITED WITHOUT A LICENSE UNDER APPLICABLE PATENTS IN THE MPEG-2 PATENT PORTFOLIO, WHICH LICENSE IS AVAILABLE FROM MPEG LA, LLC, 4600 S. Ulster Street, Suite 400, Denver, Colorado 80237 U.S.A.

5.3.2 MPEGLA MPEG4 VISUAL

THIS PRODUCT IS LICENSED UNDER THE MPEG-4 VISUAL PATENT PORTFOLIO LICENSE FOR THE PERSONAL AND NON-COMMERCIAL USE OF A CONSUMER FOR (i) ENCODING VIDEO IN COMPLIANCE WITH THE MPEG-4 VISUAL STANDARD (“MPEG-4 VIDEO”) AND/OR (ii) DECODING MPEG-4 VIDEO THAT WAS ENCODED BY A CONSUMER ENGAGED IN A PERSONAL AND NON-COMMERCIAL ACTIVITY AND/OR WAS OBTAINED FROM A VIDEO PROVIDER LICENSE IS GRANTED OR SHALL BE IMPLIED FOR ANY OTHER USE. ADDITIONAL INFORMATION INCLUDING THAT RELATING TO PROMOTIONAL, INTERNAL AND COMMERCIAL USES AND LICENSING MAY BE OBTAINED FROM MPEG LA, LLC. SEE HTTP://WWW.MPEGLA.COM.

5.3.3 MPEGLA AVC

THIS PRODUCT IS LICENSED UNDER THE AVC PATENT PORTFOLIO LICENSE FOR THE PERSONAL USE OF A CONSUMER OR OTHER USES IN WHICH IT DOES NOT RECEIVE REMUNERATION TO (i) ENCODE VIDEO IN COMPLIANCE WITH THE AVC STANDARD (“AVC VIDEO”) AND/OR (ii) DECODE AVC VIDEO THAT WAS ENCODED BY A CONSUMER ENGAGED IN A PERSONAL ACTIVITY AND/OR WAS OBTAINED FROM A VIDEO PROVIDER LICENSED TO PROVIDE AVC VIDEO. NO LICENSE IS GRANTED OR SHALL BE IMPLIED FOR ANY OTHER USE. ADDITIONAL INFORMATION MAY BE OBTAINED FROM MPEG LA, L.L.C. SEE HTTP://WWW.MPEGLA.COM.

5.3.4 MPEG4 SYSTEMS

THIS PRODUCT IS LICENSED UNDER THE MPEG-4 SYSTEMS PATENT PORTFOLIO LICENSE FOR ENCODING IN COMPLIANCE WITH THE MPEG-4 SYSTEMS STANDARD, EXCEPT THAT AN ADDITIONAL LICENSE AND PAYMENT OF ROYALTIES ARE NECESSARY FOR ENCODING IN CONNECTION WITH (i) DATA STORED OR REPLICATED IN PHYSICAL MEDIA WHICH IS PAID FOR ON A TITLE BY TITLE BASIS AND/OR (ii) DATA WHICH IS PAID FOR ON A TITLE BY TITLE BASIS AND IS TRANSMITTED TO AN END USER FOR PERMANENT STORAGE AND/OR USE.

SUCH ADDITIONAL LICENSE MAY BE OBTAINED FROM MPEG LA, LLC. SEE HTTP://WWW.MPEGLA.COM FOR ADDITIONAL DETAILS.

5.4 Drastic Technologies Limited Warranty and Disclaimers

Drastic Technologies Ltd (the Company) warrants to the original registered end user that the product will perform as stated below for a period of ninety (90) days from the date of licensing or; in the case of hardware, for a period matching the warranty period offered by the original manufacturer of said equipment.

Hardware and Media—The Product hardware components, if any, including equipment supplied but not manufactured by the Company but NOT including any third party equipment that has been substituted by the Distributor or customer for such equipment (the “Hardware”), will be free from defects in materials and workmanship under normal operating conditions and use.

5.4.1 Warranty Remedies

Your sole remedies under this limited warranty are as follows:
Hardware and Media—The Company will either repair or replace (at its option) any defective Hardware component or part, or Software Media, with new or like new Hardware components or Software Media. Components may not be necessarily the same, but will be of equivalent operation and quality.

5.4.2 Software Updates

Except as may be provided in a separate agreement between Drastic Technologies and You, if any, Drastic Technologies is under no obligation to maintain or support the Software and Drastic Technologies has no obligation to furnish you with any further assistance, technical support, documentation, software, update, upgrades, or information of any nature or kind.

5.4.3 Restrictions and Conditions of Limited Warranty

This Limited Warranty will be void and of no force and effect if (i) Product Hardware or Software Media, or any part thereof, is damaged due to abuse, misuse, alteration, neglect, or shipping, or as a result of service or modification by a party other than the Company, or (ii) Software is modified without the written consent of the Company.

5.4.4 Limitations of Warranties

THE EXPRESS WARRANTIES SET FORTH IN THIS AGREEMENT ARE IN LIEU OF ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. No oral or written information or advice given by the Company, its distributors, dealers or agents, shall increase the scope of this Limited Warranty or create any new warranties.

Geographical Limitation of Warranty—This limited warranty is valid only within the country in which the Product is purchased/licensed.

Limitations on Remedies—YOUR EXCLUSIVE REMEDIES, AND THE ENTIRE LIABILITY OF Drastic Technologies Ltd WITH RESPECT TO THE PRODUCT, SHALL BE AS STATED IN THIS LIMITED WARRANTY. Your sole and exclusive remedy for any and all breaches of any Limited Warranty by the Company shall be the recovery of reasonable damages which, in the aggregate, shall not exceed the total amount of the combined license fee and purchase price paid by you for the Product.

5.4.5 Damages

Drastic Technologies Ltd SHALL NOT BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS, OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF YOUR USE OR INABILITY TO USE THE PRODUCT, OR THE BREACH OF ANY EXPRESS OR IMPLIED WARRANTY, EVEN IF THE COMPANY HAS BEEN ADVISED OF THE POSSIBILITY OF THOSE DAMAGES, OR ANY REMEDY PROVIDED FAILS OF ITS ESSENTIAL PURPOSE.

Further information regarding this limited warranty may be obtained by writing:
Drastic Technologies Ltd
523 The Queensway, Suite 201
Toronto, ON, M8V 1J7
Telephone: (416) 255-5636

Drastic Technologies Ltd. does not assume responsibility for loss or damage resulting from errors, omissions, or inaccuracies herein. This document is subject to change, and revisions may be made and issued to include such changes.

No part of this document may be reproduced, saved to a storage and retrieval system, or transmitted in any form or by any means, electronic, mechanical, recorded, or otherwise without the prior written consent of Drastic Technologies Ltd.

This manual has been compiled to assist the user in their experience using **DrasticScope** software. It is believed to be correct at the time of writing, and every effort has been made to provide accurate and useful information. Any errors that may have crept in are unintentional and will hopefully be purged in a future revision of this document. We welcome your feedback.

Copyright 2025 © Drastic Technologies Ltd. All rights reserved. Software products licensed are owned by Drastic Technologies Ltd. and are protected by international treaty provisions and national copyright laws. All Rights Reserved.